SCALABLE FIRST-ORDER INTERIOR POINT TRUST REGION ALGORITHMS FOR LINEARLY CONSTRAINED OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Computing approximate Karush–Kuhn-Tucker (KKT) points for constrained nonconvex programs is a fundamental problem in mathematical programming. Interior-point trust-region (IPTR) methods are particularly attractive for such problems because they maintain strictly feasible iterates throughout the iterative process and converge to a first-order and second-order KKT solution. Their scalability, however, is limited by the repeated computation of trust-region search directions. In this paper, we propose an approximate first-order IPTR framework that addresses this bottleneck by replacing exact trust-region subproblem solves with an approximate projector maintained through low-rank updates. The resulting method preserves feasibility and the global convergence guarantees of standard IPTR schemes while substantially reducing the per-iteration cost. We further extend the framework to obtain approximate second-order KKT points using only first-order information by integrating a gradient-based negative-curvature routine, thus avoiding explicit Hessian computations. We conduct numerical experiments to demonstrate the scalability of our approximate first-order IPTR framework in large-scale settings, where it achieves up to a $2.80\times$ speedup over the existing first-order IPTR algorithm.

1 INTRODUCTION

In this paper, we consider the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}, \end{aligned} \tag{1}$$

where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{b} \in \mathbb{R}^m$. We assume that $m \leq n$ and that $\boldsymbol{A}$ has full row rank. The objective function $f\colon \mathbb{R}^n \to \mathbb{R}$ is continuous on the nonnegative orthant $\mathbb{R}^n_+ := \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{x} \geq \boldsymbol{0}\}$ and differentiable on the strictly positive orthant $\mathbb{R}^n_{++} := \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{x} > \boldsymbol{0}\}$. Problem (1) is a fundamental formulation in machine learning, covering a broad class of optimization problems with nonnegativity and affine equality constraints. Such problems arise in sparse regression (Xiao & Bai, 2022), multiple kernel learning for classification (Rakotomamonjy et al., 2008), and neural-network training with ℓ_∞ constraints (Xie & Li, 2024). In many such problems, the objective function f is nonconvex, making the search for global minimizers NP-hard. Consequently, theoretical and algorithmic developments typically focus on identifying local minimizers or stationary points that satisfy necessary optimality conditions.

In the unconstrained setting, such optimality conditions are well-established through the notions of first- and second-order stationarity. A point $\boldsymbol{x}$ is called an ε-first-order stationary point (FOSP) if $\|\nabla f(\boldsymbol{x})\| \leq \varepsilon$. An $(\varepsilon, \sqrt{\varepsilon})$-second-order stationary point (SOSP) further requires that $\lambda_{\min}(\nabla^2 f(\boldsymbol{x})) \geq -\sqrt{\rho\varepsilon}$, where ρ is the Lipschitz constant of the Hessian. An ε-FOSP captures first-order stationarity, but it is generally insufficient in nonconvex optimization since it may still be a saddle point. Such saddle points can be highly suboptimal and thus undesirable in practice (Jain et al., 2017; Sun et al., 2018). This motivates algorithms that go beyond first-order stationarity and provably converge to SOSPs. A direct way is to use second-order methods, but explicitly forming and factorizing the Hessian typically costs $\Omega(n^2)$, which can be prohibitive in high-dimensional

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

problems. A large body of work therefore focuses on obtaining second-order guarantees using primarily gradient information. When the gradient is small, either the Hessian is nearly positive semi-definite so $\boldsymbol{x}$ is already close to an SOSP, or a direction of negative curvature exists that allows further descent. Such directions can be found using first-order methods. In particular, prior work (Allen-Zhu & Li, 2018; Carmon et al., 2018; Liu et al., 2018; Xu et al., 2017; Jin et al., 2018; Zhang & Li, 2021) establishes that $\widetilde{\mathcal{O}}(1/\varepsilon^{1.75})$[1] gradient queries suffice to find an $(\varepsilon, \sqrt{\varepsilon})$-SOSP. Extending this $\widetilde{O}(1/\varepsilon^{1.75})$ complexity to constrained optimization remains an open question.

Constrained optimization, such as constrained learning and training in AI, has become popular and even necessary in practice (Nandwani et al., 2019; Huang et al., 2024; Qin et al., 2022; Wachi et al., 2024; Liu et al., 2024; Yu et al., 2022). However, the aforementioned stationarity-based characterization is no longer sufficient in such constrained settings, where feasibility must be taken into account and optimal solutions may lie on the boundary of the feasible region. In such cases, vanishing gradients or positive semidefinite Hessians of the objective alone do not capture optimality. The KKT conditions provide a principled extension of first- and second-order stationarity to constrained problems by jointly incorporating the objective and the constraints through primal–dual optimality conditions. Under standard regularity assumptions, KKT points are necessary for local optimality and thus play a role analogous to that of stationary points in the unconstrained setting (Bertsekas, 1997; Nocedal & Wright, 2006). KKT points are widely used in the analysis of machine learning algorithms to characterize their convergence and implicit bias. In neural-network training, they connect the limiting parameter directions of gradient descent for homogeneous networks to margin-maximization problems (Lyu & Li, 2019) and characterize the convergence limits of full-batch AdamW through ℓ_∞-constrained loss minimization (Xie & Li, 2024). This motivates the study of algorithms that converge to approximate first-order and second-order KKT points in constrained nonconvex optimization. A variety of algorithms have been proposed to identify approximate KKT points in constrained optimization, including trust-region methods (Haeser et al., 2019) and augmented Lagrangian approaches (Li & Xu, 2021; Kong et al., 2023). Inspired by unconstrained optimization, where gradient-based perturbations allow efficient escape from saddle points, it is natural to seek methods that achieve similar efficiency in the constrained setting. In particular, we aim at reaching approximate first- and second-order KKT points using only gradient information, which can avoid the heavy computational cost of second-order oracles.

Our contributions Our main contribution is a scalable first-order interior-point trust-region framework for computing approximate first- and second-order KKT points in linearly constrained nonconvex optimization. Existing IPTR methods face two major computational bottlenecks: first-order methods require repeatedly recomputing an expensive projection as the iterates change, while second-order methods additionally rely on Hessian information. We overcome both bottlenecks within a unified framework. For first-order KKT points, we replace the exact projection with an efficiently maintainable approximation, reducing the average per-iteration cost from $O(nm^{\omega-1})$ to the matrix-vector multiplication time $\widetilde{O}(nm)$. For second-order KKT points, we develop a feasibility-preserving gradient-based negative-curvature procedure that avoids Hessian evaluations and enables the first-order IPTR framework to escape saddle points and converge to $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 points. Table 1 summarizes the oracle requirements, iteration bounds, and runtime complexities of our algorithms and the existing IPTR methods.

Techniques In existing IPTR methods, the search direction for (1) at each iteration is obtained by minimizing a local model of the potential function over a trust-region ball, while maintaining feasibility with respect to the linear equality constraints:

$$\min_{\boldsymbol{d}} \begin{cases} \nabla\phi(\boldsymbol{x}_t)^\top \boldsymbol{X}_t \boldsymbol{d} & \text{first-order IPTR} \\ \nabla\phi(\boldsymbol{x}_t)^\top \boldsymbol{X}_t \boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^\top \boldsymbol{X}_t \nabla^2 f(\boldsymbol{x}_t) \boldsymbol{X}_t \boldsymbol{d} & \text{second-order IPTR} \end{cases} \quad \text{s.t. } \boldsymbol{A}\boldsymbol{X}_t\boldsymbol{d} = \boldsymbol{0},\ \|\boldsymbol{d}\| \le \beta; \tag{2}$$

where $\boldsymbol{X}_t := \mathrm{diag}(\boldsymbol{x}_t)$, $\phi(\boldsymbol{x}_t)$ is the potential function, $\boldsymbol{A}$ is the constraint matrix in (1), and $\beta < 1$ is the trust-region radius.

While the first- and second-order IPTR methods can converge to approximate first- and second-order KKT points respectively, they both suffer from high computational complexity in large-

[1]Throughout this paper, we use $\widetilde{O}(\cdot)$ to suppress polylogarithmic factors in ε^{-1} and n.

Table 1: Iteration and overall time complexities of IPTR-type algorithms for computing approximate first- and second-order KKT points.

Criteria	Algorithm	Oracle	Iterations	Time complexity
2ε-KKT	Haeser et al. (2019)	1st-order	$\mathcal{O}(\varepsilon^{-2})$	$\mathcal{O}\left(\frac{nm^{\omega-1}}{\varepsilon^2}\right)$
2ε-KKT	Algorithm 1	1st-order	$\mathcal{O}(\varepsilon^{-2})$	$\widetilde{\mathcal{O}}\left(nm^{\omega-1}+\frac{nm}{\varepsilon^2}\right)$
2ε-KKT (concave f)	Algorithm 1	1st-order	$\mathcal{O}(\varepsilon^{-1})$	$\widetilde{\mathcal{O}}\left(nm^{\omega-1}+\min\left\{\frac{nm}{\varepsilon^2},\frac{nm^{\omega-1}}{\varepsilon}\right\}\right)$
$(2\varepsilon,\sqrt{\varepsilon})$-KKT2	Haeser et al. (2019)	2nd-order	$\mathcal{O}(\varepsilon^{-1.5})$	$\widetilde{\mathcal{O}}\left(\frac{n^{\omega}}{\varepsilon^{1.5}}\right)$
$(2\varepsilon,\sqrt{\varepsilon})$-KKT2	Algorithm 2	1st-order	$\widetilde{\mathcal{O}}(\varepsilon^{-2})$	$\widetilde{\mathcal{O}}\left(\frac{nm^{\omega-1}}{\varepsilon^2}\right)$
$(2\varepsilon,\sqrt{\varepsilon})$-KKT2	Algorithm 3	1st-order	$\widetilde{\mathcal{O}}(\varepsilon^{-2})$	$\widetilde{\mathcal{O}}\left(\frac{nm^{\omega-1}}{\varepsilon^{1.5}}+\frac{nm}{\varepsilon^2}\right)$

scale settings. The first-order IPTR subproblem admits a closed-form solution given by $-\beta \boldsymbol{P}_t \boldsymbol{X}_t \nabla\phi(\boldsymbol{x}_t)/\|\boldsymbol{P}_t \boldsymbol{X}_t \nabla\phi(\boldsymbol{x}_t)\|$, where $\boldsymbol{P}_t$ denotes the projection in the null space of $\boldsymbol{A}\boldsymbol{X}_t$. A major computational bottleneck of this step is that computing the search direction requires an explicit basis for the null space of $\boldsymbol{A}\boldsymbol{X}_t$. On the other hand, the second-order IPTR requires access to the exact Hessian $\nabla^2 f(\boldsymbol{x}_t)$, which incurs at least $\Omega(n^2)$ time and space complexity per iteration. Since $\boldsymbol{x}_t$ varies across iterations, continuously recomputing the projection matrix and the Hessian becomes computationally expensive.

To overcome these computational barriers, we first address the projection cost by developing an approximate first-order IPTR algorithm that avoids recomputing the exact projection matrix from scratch. When $\boldsymbol{A}$ has full row rank, the projection matrix takes the explicit form $\boldsymbol{P}_t := \boldsymbol{I} - \boldsymbol{X}_t\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{X}_t^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}_t$. The main computational cost arises from forming and inverting the matrix $\boldsymbol{A}\boldsymbol{X}_t^2\boldsymbol{A}^\top$. To mitigate this cost, we maintain an approximate projector $\boldsymbol{R}_t := \boldsymbol{I} - \boldsymbol{X}_t^{-1}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}_t$, where $\overline{\boldsymbol{X}}_t$ is a maintained approximation of $\boldsymbol{X}_t$. The matrix $\overline{\boldsymbol{X}}_t$ is updated sparsely using the binary-decomposition scheme. This sparsity structure enables efficient low-rank Sherman–Morrison–type updates of $(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}$. The resulting projector preserves feasibility of the search directions while reducing the computational cost through structured low-rank updates.

To obtain approximate second-order KKT points using only first-order information, we develop a feasibility-preserving negative-curvature procedure (Procedure 1). This differs from standard negative-curvature finding in unconstrained optimization, since both the relevant Hessian and the feasible subspace depend on the current iterate $\boldsymbol{x}_t$, and the gradient queries must remain in the interior feasible region. Our procedure applies projected power iteration on the feasible subspace and approximates Hessian–vector products using finite differences of gradients. If the second-order condition is violated, it finds a feasible negative-curvature direction that decreases the potential function by $\Omega(\varepsilon^{3/2})$; otherwise, the current iterate satisfies the $(2\varepsilon,\sqrt{\varepsilon})$-KKT2 conditions.

Numerical experiments We also perform extensive numerical experiments. Through a variety of test instances, we show that our first-order IPTR algorithms are able to compute second-order approximate KKT points without access to Hessian information. In addition, we test our approximate first-order IPTR algorithms on large-scale optimization problems with $(n,m) = (1000,500)$, $(3000,2000)$ and $(5000,4000)$ of the problem in Eq. (1) to evaluate their practical efficiency. The experimental results show that our approximate IPTR algorithm achieves up to a $2.80\times$ speedup over the existing first-order IPTR algorithm on these instances, demonstrating its scalability for large-scale problems.

Paper organization In Section 2, we introduce the notions of approximate first-order and second-order KKT points and state the assumptions. In Section 3, we present an approximate first-order IPTR algorithm with a computationally efficient low-rank update. In Section 4, we introduce a negative-curvature finding procedure that enables the first-order IPTR framework to converge to approximate second-order KKT points. Numerical experiments are reported in Section 5.

2 PRELIMINARIES

Throughout this paper, we use boldface letters to denote matrices and vectors. For any vector $\boldsymbol{x}$, we use $\boldsymbol{X} := \mathrm{diag}\{\boldsymbol{x}\}$ to denote the diagonal matrix with $\boldsymbol{x}$ on its diagonal. We define the feasible set as $\Omega := \{\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\}$, and its interior as $\Omega^\circ = \{\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} > \boldsymbol{0}\}$. Unless stated otherwise, $\|\cdot\|$ denotes the Euclidean norm.

2.1 APPROXIMATE KKT POINTS AND SUFFICIENT CONDITIONS

We consider linearly constrained optimization problems where the objective function is continuous on the nonnegative orthant but may lack first- or second-order differentiability on the boundary. Standard first- and second-order KKT conditions may thus fail to be well-defined at a boundary local solution. Meanwhile, interior-point methods generate strictly interior iterates that only approach the boundary in the limit. This motivates approximate optimality criteria that can be evaluated at interior feasible points while still characterizing convergence toward boundary KKT points. We adopt the perturbed KKT conditions in Haeser et al. (2019), which, for linearly constrained problems, yield the following sufficient conditions for approximate first- and second-order KKT points.

Proposition 1 (Haeser et al. (2019, Proposition 1))**.** *Let $\varepsilon > 0$. A point $\boldsymbol{x}$ is said to satisfy the sufficient condition for being an ε-KKT point if there exists a vector $\boldsymbol{v} \in \mathbb{R}^m$ such that:*

(i) $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{x} > 0$
(ii) $\nabla f(\boldsymbol{x}) + \boldsymbol{A}^\top \boldsymbol{v} \geq -\varepsilon$
(iii) $\|\boldsymbol{X}(\nabla f(\boldsymbol{x}) + \boldsymbol{A}^\top \boldsymbol{v})\|_\infty \leq \varepsilon$

Proposition 2 (Haeser et al. (2019, Proposition 2))**.** *Let $\varepsilon_1, \varepsilon_2 > 0$. A point $\boldsymbol{x}$ satisfies the sufficient condition for being an $(\varepsilon_1, \varepsilon_2)$-KKT2 point if there exists a vector $\boldsymbol{v} \in \mathbb{R}^m$ such that:*

(i) $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{x} > 0$
(ii) $\nabla f(\boldsymbol{x}) + \boldsymbol{A}^\top \boldsymbol{v} \geq -\varepsilon_1$
(iii) $\|\boldsymbol{X}(\nabla f(\boldsymbol{x}) + \boldsymbol{A}^\top \boldsymbol{v})\|_\infty \leq \varepsilon_1$
(iv) $\lambda_{\min}(\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X})_{\boldsymbol{A}\boldsymbol{X}} \geq -\varepsilon_2$

The lower bound on $\nabla f(\boldsymbol{x}) + \boldsymbol{A}^\top \boldsymbol{v}$ controls approximate stationarity, while $\boldsymbol{X}(\nabla f(\boldsymbol{x}) + \boldsymbol{A}^\top \boldsymbol{v})$ acts as an approximate complementarity residual. Similarly, the second-order condition measures curvature along the scaled feasible directions $\boldsymbol{d} \in \ker(\boldsymbol{A}\boldsymbol{X})$.

2.2 ASSUMPTIONS

This section details the assumptions imposed on the objective function. Rather than assuming global Lipschitz continuity of the gradient or the Hessian in the original variable $\boldsymbol{x}$, we require Lipschitz-type bounds only along the locally feasible displacement directions $\boldsymbol{d}$. These displacements are measured in the scaled coordinate system defined by the diagonal matrix $\boldsymbol{X}$ and restricted to satisfy the linearized feasibility condition $\boldsymbol{A}\boldsymbol{X}\boldsymbol{d} = 0$. The following assumptions formalize these scaled Lipschitz properties for the gradient and the Hessian.

Assumption 1. *Suppose $f(\boldsymbol{x})$ is twice differentiable on Ω°. Then, for any $\boldsymbol{x} \in \Omega^\circ$ and any $\boldsymbol{d}, \boldsymbol{d}' \in \{\boldsymbol{d} : \|\boldsymbol{d}\| \leq \gamma,\ \boldsymbol{A}\boldsymbol{X}\boldsymbol{d} = \boldsymbol{0}\}$, where $\gamma < 1$ is a fixed constant, the following conditions hold:*

$$\|\boldsymbol{X}\left(\nabla f(\boldsymbol{X}(\boldsymbol{e} + \boldsymbol{d})) - \nabla f(\boldsymbol{X}(\boldsymbol{e} + \boldsymbol{d}'))\right)\| \leq l\|\boldsymbol{d} - \boldsymbol{d}'\|. \tag{3}$$

Assumption 2. *Suppose $f(\boldsymbol{x})$ is twice differentiable on Ω°. Then, for any $\boldsymbol{x} \in \Omega^\circ$ and any $\boldsymbol{d}, \boldsymbol{d}' \in \{\boldsymbol{d} : \|\boldsymbol{d}\| \leq \gamma,\ \boldsymbol{A}\boldsymbol{X}\boldsymbol{d} = \boldsymbol{0}\}$, where $\gamma < 1$ is a fixed constant, the following conditions hold:*

$$\|\boldsymbol{X}\left(\nabla^2 f(\boldsymbol{X}(\boldsymbol{e} + \boldsymbol{d})) - \nabla^2 f(\boldsymbol{X}(\boldsymbol{e} + \boldsymbol{d}'))\right)\boldsymbol{X}\| \leq \rho\|\boldsymbol{d} - \boldsymbol{d}'\|. \tag{4}$$

All first-order algorithms presented in this paper rely on Assumption 1. When such first-order methods are augmented with a negative-curvature finding procedure to obtain second-order KKT points, an additional Assumption 2 is required.

For $\varepsilon > 0$, we define the logarithmic-barrier potential function $\phi(\boldsymbol{x}) = f(\boldsymbol{x}) - \varepsilon \sum_{i=1}^n \ln(x_i)$.

Assumption 3. *There exists a constant $L_\phi > 0$ such that $\|\boldsymbol{X}\nabla\phi(\boldsymbol{x})\| \leq L_\phi$ for all $\boldsymbol{x} \in \Omega^\circ$.*

Assumption 3 requires the scaled gradient of the potential function to remain bounded over the interior feasible region. Since $\boldsymbol{X}\nabla\phi(\boldsymbol{x}) = \boldsymbol{X}\nabla f(\boldsymbol{x}) - \varepsilon\boldsymbol{e}$, this condition does not require the unscaled gradient $\nabla f(\boldsymbol{x})$ itself to be uniformly bounded. It allows the gradient to grow as an iterate approaches the boundary, provided that this growth is controlled under the interior-point scaling.

3 Approximate First-order Interior Point Trust Region Algorithm

We now develop an approximate first-order IPTR algorithm that reduces the computational cost of the standard first-order IPTR method while preserving its convergence guarantee. We first recall the exact first-order IPTR step and identify its main computational bottleneck.

At iteration t, the standard first-order IPTR method computes a search direction by solving the trust-region subproblem

$$\begin{aligned} \min \quad & \nabla\phi(\boldsymbol{x}_t)^\top \boldsymbol{X}_t \boldsymbol{d} \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{X}_t\boldsymbol{d} = 0,\ \|\boldsymbol{d}\| \le \beta, \end{aligned} \tag{5}$$

where β denotes the trust-region radius. When $\boldsymbol{P}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t) \neq \boldsymbol{0}$, the exact solution to (5) is

$$\boldsymbol{d}_t := -\beta \frac{\boldsymbol{P}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)}{\|\boldsymbol{P}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|}, \tag{6}$$

where $\boldsymbol{P}_t \coloneqq \boldsymbol{I} - \boldsymbol{X}_t\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{X}_t^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}_t$ is the orthogonal projector onto $\ker(\boldsymbol{A}\boldsymbol{X}_t)$. The next iterate is then updated as $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{X}_t\boldsymbol{d}_t$. The main computational bottleneck is evaluating the projection matrix $\boldsymbol{P}_t$. Since $\boldsymbol{X}_t$ changes at every iteration, the matrix $\boldsymbol{A}\boldsymbol{X}_t^2\boldsymbol{A}^\top$ also changes, and recomputing its inverse from scratch incurs $\mathcal{O}(nm^{\omega-1})$ time per iteration. This repeated computation becomes expensive for large-scale problems.

Our Algorithm 1 overcomes this bottleneck by reducing the average per-iteration complexity to $\mathcal{O}(mn)$. This makes each iteration as cheap as a single matrix-vector multiplication. The key idea is to replace the frequently changing exact diagonal $\boldsymbol{X}_t$ in the inversion term $(\boldsymbol{A}\boldsymbol{X}_t^2\boldsymbol{A}^\top)^{-1}$ with a sparsely updated diagonal approximation $\overline{\boldsymbol{X}}_t$. This yields an approximate projection matrix $\boldsymbol{R}_t$:

$$\boldsymbol{R}_t \coloneqq \boldsymbol{I} - \boldsymbol{X}_t^{-1}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}_t. \tag{7}$$

This specific construction of $\boldsymbol{R}_t$ not only ensures search direction strictly resides within null space of $\boldsymbol{A}\boldsymbol{X}_t$, but also allows rapid updates since the diagonal matrix $\overline{\boldsymbol{X}}_t$ is modified sparsely. Based on this efficient projection, we scale the step to a norm of β whenever projected gradient is non-zero, and set it to zero otherwise. Accordingly, the approximate search direction $\widetilde{\boldsymbol{d}}_t$ is computed as

$$\widetilde{\boldsymbol{d}}_t := \begin{cases} \boldsymbol{0}, & \text{if } \boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t) = \boldsymbol{0}, \\ -\beta\frac{\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)}{\|\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|}, & \text{otherwise.} \end{cases} \tag{8}$$

It remains to maintain $\overline{\boldsymbol{X}}_t$ so that it stays close to $\boldsymbol{X}_t$ while changing only sparsely across iterations. Since $\boldsymbol{x}_{t+1} = \boldsymbol{X}_t(\boldsymbol{e} + \widetilde{\boldsymbol{d}}_t)$ and $\|\widetilde{\boldsymbol{d}}_t\| \le \beta$, the relative change of the iterate between consecutive iterations is small. We therefore maintain $\overline{\boldsymbol{X}}_t$ as a lazy approximation of $\boldsymbol{X}_t$, updating only coordinates whose accumulated relative changes become sufficiently large. Specifically, we apply the `SelectVector` algorithm from Lee & Vempala (2021, Algorithm 4) to the logarithms of the iterates. The following lemma guarantees that $\overline{\boldsymbol{x}}_t$ remains close to $\boldsymbol{x}_t$ while only a small number of coordinates are updated at each iteration.

Lemma 1 (Lee & Vempala (2021, Lemma 19)). *Given vectors $\ln\boldsymbol{x}_0, \ln\boldsymbol{x}_1, \ln\boldsymbol{x}_2, \ldots$ arriving in a stream, and satisfies that $\|\ln\boldsymbol{x}_{t+1} - \ln\boldsymbol{x}_t\|_2 \le 2\beta$ for all t. For any $\frac{1}{2} > \delta > 0$, define the vector $\ln\overline{\boldsymbol{x}}_0 = \ln\boldsymbol{x}_0$ and $\ln\overline{\boldsymbol{x}}_t = \texttt{SelectVector}(\ln\overline{\boldsymbol{x}}_{t-1}, \ln\boldsymbol{x}_0, \ln\boldsymbol{x}_1, \ldots, \ln\boldsymbol{x}_t, \delta)$. Then, we have that*

(i) $\|\ln\overline{\boldsymbol{x}}_t - \ln\boldsymbol{x}_t\|_\infty \le \delta$ *for all k.*

(ii) $\|\ln\overline{\boldsymbol{x}}_t - \ln\overline{\boldsymbol{x}}_{t-1}\|_0 \le O(2^{2l_t}(2\beta/\delta)^2\log^2 n)$ *where l_t is the largest integer l with $t = 0 \mod 2^l$.*

By Lemma 1(i), $e^{-\delta}\boldsymbol{x}_t \le \overline{\boldsymbol{x}}_t \le e^{\delta}\boldsymbol{x}_t$, so $\overline{\boldsymbol{X}}_t$ remains a multiplicative approximation of $\boldsymbol{X}_t$. Lemma 1(ii) shows that only a small number of coordinates of $\overline{\boldsymbol{X}}_t$ change between consecutive iterations. Thus, $\boldsymbol{A}\overline{\boldsymbol{X}}_{t+1}^2\boldsymbol{A}^\top$ is a low-rank update of $\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top$, and its inverse can be updated using the Sherman–Morrison–Woodbury formula instead of being recomputed from scratch.

The complete algorithm is summarized in Algorithm 1. We initialize the algorithm at an approximate analytic center $\boldsymbol{x}_0 \in \Omega^\circ$ satisfying $-\sum_{i=1}^n \ln(x_i) \ge -\sum_{i=1}^n \ln(x_{0,i}) - C_0$ for all $\boldsymbol{x} \in \Omega^\circ$, and set $\overline{\boldsymbol{x}}_0 = \boldsymbol{x}_0$. At iteration t, it approximately solves the subproblem (5) using the projection matrix

$\boldsymbol{R}_t$, computes the direction $\widetilde{\boldsymbol{d}}_t$ by (8), and updates $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{X}_t\widetilde{\boldsymbol{d}}_t$. It then checks whether the potential function decreases sufficiently. If not, the algorithm returns $\boldsymbol{x}_t$. Otherwise, it updates $\overline{\boldsymbol{x}}_{t+1}$ by applying `SelectVector` to the logarithms of the iterates.

Algorithm 1: Approximate First-order Interior Point Trust Region Algorithm

Initialize $\boldsymbol{x}_0$ as an approximate analytic center and set $\overline{\boldsymbol{x}}_0 \leftarrow \boldsymbol{x}_0$;
$T \leftarrow \frac{(f(\boldsymbol{x}_0)-f(\boldsymbol{x}^*)+(C_0-1)\varepsilon)(l+2\varepsilon+2)}{\varepsilon^2}$;
$\beta \leftarrow \varepsilon/(l+2\varepsilon+2), \quad \delta \leftarrow \min\{\varepsilon/(15L_\phi), \beta/(92L_\phi)\}$;
for $t = 0, \ldots, T-1$ **do**
  $\boldsymbol{R}_t \leftarrow \boldsymbol{I} - \boldsymbol{X}_t^{-1}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}_t$, $\widetilde{\boldsymbol{d}}_t \leftarrow \boldsymbol{0}$;
  if $\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t) \neq \boldsymbol{0}$ **then** $\widetilde{\boldsymbol{d}}_t \leftarrow -\beta\frac{\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)}{\|\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|}$;
  $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t + \boldsymbol{X}_t\widetilde{\boldsymbol{d}}_t$;
  if $\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) > -\varepsilon^2/(2l+4\varepsilon+4)$ **then return** $\boldsymbol{x}_t$;
  $\ln\overline{\boldsymbol{x}}_{t+1} \leftarrow$ `SelectVector`$(\ln\overline{\boldsymbol{x}}_t, \ln\boldsymbol{x}_0, \ldots, \ln\boldsymbol{x}_{t+1}, \delta)$;

We first establish the time complexity of Algorithm 1. The saving comes from maintaining $(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}$ instead of recomputing it every iteration. The following lemma bounds the cost of a single inverse update.

Lemma 2. *Let $\|\overline{\boldsymbol{x}}_{t+1} - \overline{\boldsymbol{x}}_t\|_0 = q_t$. Given $(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}$, the inverse $(\boldsymbol{A}\overline{\boldsymbol{X}}_{t+1}^2\boldsymbol{A}^\top)^{-1}$ can be updated in time $\mathcal{O}(m^2q_t^{\omega-2})$ when $q_t \leq m$, and in time $\mathcal{O}(nm^{\omega-1})$ when $m < q_t \leq n$, where ω denotes the exponent of matrix multiplication.*

Summing the inverse-update cost over all iterations according to the number of updated coordinates q_t gives the overall time complexity.

Proposition 3. *Suppose Algorithm 1 runs for T iterations, and let $\delta = \Theta(\beta)$. Then, the overall time complexity is upper bounded by $\widetilde{\mathcal{O}}\left(nm^{\omega-1} + mnT\right)$.*

We next establish the stopping criterion and convergence guarantee of Algorithm 1. Under the multiplicative approximation $e^{-\delta}\boldsymbol{X}_t \leq \overline{\boldsymbol{X}}_t \leq e^{\delta}\boldsymbol{X}_t$, the approximate projector satisfies $\|\boldsymbol{R}_t - \boldsymbol{P}_t\| \leq 46\delta$ in Lemma 4. Using this bound, we show that each iteration either yields sufficient decrease in the potential function or satisfies the desired approximate first-order KKT conditions.

Proposition 4. *Under Assumption 1 and Assumption 3, for any $\varepsilon \in (0, \min\{\gamma, \frac{1}{2}\}]$, let $\beta = \varepsilon/(l+2\varepsilon+2)$ and $\delta = \min(\varepsilon/(15L_\phi), \beta/(92L_\phi))$. Suppose that $e^{-\delta}\boldsymbol{x}_t \leq \overline{\boldsymbol{x}}_t \leq e^{\delta}\boldsymbol{x}_t$, and define the next iterate by $\boldsymbol{x}_{t+1} := \boldsymbol{X}_t(\boldsymbol{e} + \widetilde{\boldsymbol{d}}_t)$, where $\widetilde{\boldsymbol{d}}_t$ is given in (8). Then, at iteration t, one of the following holds:*

$$\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) \leq -\frac{\varepsilon^2}{2l+4\varepsilon+4}, \tag{9}$$

or there exists $\boldsymbol{v}_t \in \mathbb{R}^m$ such that

$$\|\boldsymbol{X}_t(\nabla f(\boldsymbol{x}_t) + \boldsymbol{A}^\top\boldsymbol{v}_t)\|_\infty < 2\varepsilon, \quad \text{and} \quad \nabla f(\boldsymbol{x}_t) + \boldsymbol{A}^\top\boldsymbol{v}_t > 0. \tag{10}$$

Proposition 4 provides the stopping criterion for Algorithm 1: if the potential decreases by at least $\varepsilon^2/(2l+4\varepsilon+4)$, the algorithm proceeds; otherwise, the current iterate is a 2ε-KKT point. This sufficient decrease bounds the number of iterations, while Proposition 3 bounds the computational cost. Combining the two gives the following convergence and time complexity guarantee.

Theorem 1. *Suppose that Assumption 1 and Assumption 3 hold. For any $\varepsilon \in (0, \min\{\gamma, \frac{1}{2}\}]$, the Algorithm 1 is guaranteed to find a 2ε-KKT point within $\mathcal{O}(l(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*))/\varepsilon^2)$ gradient evaluations. Otherwise, it holds that $f(\boldsymbol{x}_t) - f(\boldsymbol{x}^*) \leq \varepsilon$. Moreover, the overall time complexity of the algorithm is $\widetilde{\mathcal{O}}\left(nm^{\omega-1} + nm/\varepsilon^2\right)$.*

Corollary 1. *Suppose f is concave on Ω° and Assumption 3 holds. Let $\varepsilon \in (0, 1]$. Consider Algorithm 1 with the trust-region radius $\beta \in (0, 1)$ and $\delta = (1-\beta)\varepsilon/(184L_\phi)$. Then, the algorithm terminates within $\mathcal{O}(1/\varepsilon)$ iterations, returning either a 2ε-KKT point or an iterate $\boldsymbol{x}_t$ satisfying $f(\boldsymbol{x}_t) - f(\boldsymbol{x}^*) \leq \varepsilon$. Furthermore, the overall time complexity is $\widetilde{\mathcal{O}}\left(nm^{\omega-1} + \min\left\{nm/\varepsilon^2, nm^{\omega-1}/\varepsilon\right\}\right)$.*

When f is concave on Ω°, the concavity inequality $f(\boldsymbol{y}) \le f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^\top(\boldsymbol{y}-\boldsymbol{x})$ yields a stronger descent guarantee, thereby improving the iteration complexity. Proofs of all results in this section are deferred to Appendix B.

4 Finding Second-Order KKT Points by First-Order Methods

In this section, we show how to find approximate second-order KKT points using solely first-order information. Compared with a first-order KKT point, an approximate second-order KKT point additionally requires the minimum eigenvalue of the scaled Hessian $\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}$ restricted to $\ker(\boldsymbol{AX})$ to be bounded below. Specifically, if $\lambda_{\min}\{\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\}_{\boldsymbol{AX}} < -\sqrt{\varepsilon}$, our goal is to find a unit negative-curvature direction $\widehat{\boldsymbol{e}} \in \ker(\boldsymbol{AX})$ satisfying $\widehat{\boldsymbol{e}}^\top \boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\widehat{\boldsymbol{e}} \le -\sqrt{\varepsilon}/4$. In the following, we let $\boldsymbol{P}$ denote the orthogonal projector onto $\ker(\boldsymbol{AX})$.

Procedure 1: Negative Curvature Finding($\boldsymbol{x}$)

$\boldsymbol{y}_0 \leftarrow$ Uniform ball with radius r centered at $\boldsymbol{x}$ in the $\ker(\boldsymbol{AX})$;
for $t = 0, \dots, \mathcal{T}-1$ **do**
  $\boldsymbol{y}_{t+1} \leftarrow \boldsymbol{y}_t - \frac{\|\boldsymbol{y}_t\|}{lr}\boldsymbol{PX}\left(\nabla f(\boldsymbol{X}(\boldsymbol{e} + \frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|})) - \nabla f(\boldsymbol{x})\right)$;

The Procedure 1 can be viewed as a power method for identifying a direction associated with the minimum eigenvalue. Indeed, since $\nabla f(\boldsymbol{X}(\boldsymbol{e} + \frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|})) - \nabla f(\boldsymbol{x}) \approx \nabla^2 f(\boldsymbol{x})\boldsymbol{X}\frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|}$, the update satisfies $\boldsymbol{y}_{t+1} \approx \left(I - \frac{1}{l}\boldsymbol{PX}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\right)\boldsymbol{y}_t$. Therefore, the iteration amplifies the component corresponding to sufficiently negative eigenvalues. Proposition 5 shows that Procedure 1 identifies a negative curvature direction with high probability.

Proposition 5. *Under Assumption 1 and Assumption 2, if* $\lambda_{\min}\{\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\}_{\boldsymbol{AX}} < -\sqrt{\varepsilon}$, *then Procedure 1 with parameters* $\dim(\ker(\boldsymbol{AX})) = k$, $\mathcal{T} = \frac{8l}{\sqrt{\varepsilon}}\log\left(\frac{8l}{\delta_0}\sqrt{\frac{n}{\pi\varepsilon}}\right)$ *and* $r = (\frac{1}{2} + \frac{7\sqrt{\varepsilon}}{16l})^{\mathcal{T}}(8\rho)^{-1}\sqrt{\frac{\pi\varepsilon}{k}}\delta_0$ *finds a negative curvature* $\widehat{\boldsymbol{e}}$ *satisfying* $\boldsymbol{AX}\widehat{\boldsymbol{e}} = 0$ *and* $\widehat{\boldsymbol{e}}^\top\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\widehat{\boldsymbol{e}} \le -\sqrt{\varepsilon}/4$ *with probability at least* $1-\delta_0$.

Once the negative curvature direction is found, the next step is to use it to escape the current saddle point of the Lagrangian. Proposition 6 demonstrates that taking a step along this negative curvature direction guarantees a sufficient decrease in the potential function.

Proposition 6. *Under Assumption 1 and Assumption 2, for any* $\varepsilon \in (0, \min\{1/100, 64\rho^2\gamma^2/9\}]$ *and any point* $\boldsymbol{x} \in \Omega^\circ$, *if there exists a unit vector* $\widehat{\boldsymbol{e}}$ *such that* $\boldsymbol{AX}\widehat{\boldsymbol{e}} = \boldsymbol{0}$ *and* $\widehat{\boldsymbol{e}}^\top\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\widehat{\boldsymbol{e}} \le -\sqrt{\varepsilon}/4$, *then the following holds:*

$$\phi\left(\boldsymbol{x} - \frac{\langle\nabla\phi(\boldsymbol{x}), \boldsymbol{X}\widehat{\boldsymbol{e}}\rangle}{|\langle\nabla\phi(\boldsymbol{x}), \boldsymbol{X}\widehat{\boldsymbol{e}}\rangle|}\cdot\frac{3\sqrt{\varepsilon}}{8\rho}\boldsymbol{X}\widehat{\boldsymbol{e}}\right) \le \phi(\boldsymbol{x}) - \frac{\sqrt{\varepsilon^3}}{1024\rho^2}. \tag{11}$$

Proposition 5 and Proposition 6 show that whenever second-order condition is violated, Procedure 1 yields a feasible direction along which the potential function decreases by $\Omega(\varepsilon^{3/2})$. This sufficient decrease ensures that the algorithm will not be trapped at first-order KKT point and will eventually converge to approximate second-order KKT point. Proofs in this section are deferred to Appendix C.

4.1 Exact First-order IPTR with Negative Curvature Finding

We now combine the negative curvature finding procedure with the exact first-order IPTR algorithm and prove that the resulting Algorithm 2, based solely on first-order information, finds an $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 point within $\mathcal{O}(1/\varepsilon^2)$ iterations and time complexity $\widetilde{\mathcal{O}}(nm^{\omega-1}/\varepsilon^2)$.

Theorem 2. *Under Assumption 1 and Assumption 2, for any* $\varepsilon \in (0, \min\{\gamma, 1/100, 64\rho^2\gamma^2/9\}]$ *and any* $\delta \in (0,1)$, *Algorithm 2 finds a* $(2\varepsilon, \sqrt{\varepsilon})$*-KKT2 point within* $\widetilde{\mathcal{O}}(l\rho^2(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*))/\varepsilon^2)$ *gradient queries, with probability at least* $1-\delta$. *The overall time complexity of the algorithm is upper bounded by* $\widetilde{\mathcal{O}}(nm^{\omega-1}/\varepsilon^2)$.

Algorithm 2: First-order IPTR with Negative Curvature Finding

Initialize $\boldsymbol{x}_0$ as an approximate analytic center;
$T \leftarrow \max\{2048(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*))\rho^2/\sqrt{\varepsilon^3}, 16\,(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + (C_0 - 1)\varepsilon)\,(l+\varepsilon)/\varepsilon^2\}$;
for $t = 0, \ldots, T-1$ **do**
  Solve (5) with $\beta = \frac{\varepsilon}{l+2\varepsilon}$ to obtain $\boldsymbol{d}_t$;
  $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t + \boldsymbol{X}_t \boldsymbol{d}_t$;
  if $\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) > -\frac{\varepsilon^2}{4l+4\varepsilon}$ **then**
    $\widehat{\boldsymbol{e}} \leftarrow$ `NegativeCurvatureFinding`$(\boldsymbol{x}_t)$;
    $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t - \frac{\langle \nabla\phi(\boldsymbol{x}), \boldsymbol{X}_t\widehat{\boldsymbol{e}}\rangle}{|\langle \nabla\phi(\boldsymbol{x}), \boldsymbol{X}_t\widehat{\boldsymbol{e}}\rangle|}\frac{3\sqrt{\varepsilon}}{8\rho}\boldsymbol{X}_t\widehat{\boldsymbol{e}}$;
    if $\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) > -\frac{\sqrt{\varepsilon^3}}{1024\rho^2}$ **then return** $\boldsymbol{x}_t$;

4.2 Approximate First-order IPTR with Negative Curvature Finding

To further reduce the overall time complexity, we combine the approximate IPTR framework (Algorithm 1) with the negative-curvature-finding procedure (Procedure 1). The main idea is that the costly exact projection is needed only when the algorithm arrives at an approximate first-order KKT point and seeks a direction of negative curvature. Since Procedure 1 is invoked at most $\mathcal{O}(1/\varepsilon^{1.5})$ times, as established in Theorem 2, we may use the cheaper approximate projection in most of the $\mathcal{O}(1/\varepsilon^2)$ iterations. This leads to Algorithm 3, which guarantees convergence to a $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 point while substantially reducing the overall time complexity.

Algorithm 3: Approximate First-order IPTR with Negative Curvature Finding

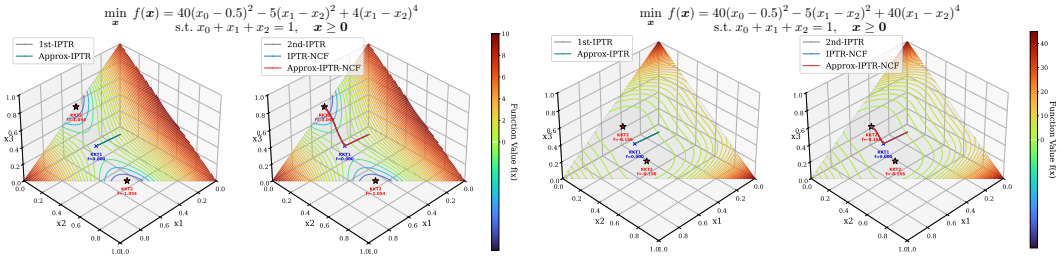
Initialize $\boldsymbol{x}_0$ as an approximate analytic center and set $\overline{\boldsymbol{x}}_0 \leftarrow \boldsymbol{x}_0$;
$T \leftarrow \max\{2048(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*))\rho^2/\sqrt{\varepsilon^3}, 4\,(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + (C_0 - 1)\varepsilon)\,(l+2\varepsilon+2)/\varepsilon^2\}$;
$\beta \leftarrow \varepsilon/(l+2\varepsilon+2), \quad \delta_{\text{err}} \leftarrow \min\{\varepsilon/(15L_\phi), \beta/(92L_\phi)\}$;
for $t = 0, \ldots, T-1$ **do**
  $\boldsymbol{R}_t \leftarrow \boldsymbol{I} - \boldsymbol{X}_t^{-1}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}_t$, $\widetilde{\boldsymbol{d}}_t \leftarrow \boldsymbol{0}$;
  if $\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t) \neq \boldsymbol{0}$ **then** $\widetilde{\boldsymbol{d}}_t \leftarrow -\beta\frac{\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)}{\|\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|}$;
  $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t + \boldsymbol{X}_t\widetilde{\boldsymbol{d}}_t$;
  if $\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) > -\frac{\varepsilon^2}{2l+4\varepsilon+4}$ **then**
    $\widehat{\boldsymbol{e}} \leftarrow$ `NegativeCurvatureFinding`$(\boldsymbol{x}_t)$;
    $\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t - \frac{\langle \nabla\phi(\boldsymbol{x}), \boldsymbol{X}_t\widehat{\boldsymbol{e}}\rangle}{|\langle \nabla\phi(\boldsymbol{x}), \boldsymbol{X}_t\widehat{\boldsymbol{e}}\rangle|}\frac{3\sqrt{\varepsilon}}{8\rho}\boldsymbol{X}_t\widehat{\boldsymbol{e}}$;
    if $\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) > -\frac{\sqrt{\varepsilon^3}}{1024\rho^2}$ **then return** $\boldsymbol{x}_t$;
  $\ln\overline{\boldsymbol{x}}_{t+1} \leftarrow$ `SelectVector`$(\ln\overline{\boldsymbol{x}}_t, \ln\boldsymbol{x}_0, \ln\boldsymbol{x}_1, \ldots, \ln\boldsymbol{x}_{t+1}, \delta_{\text{err}})$;

Theorem 3. *Suppose that Assumption 1, Assumption 2, and Assumption 3 hold. For any $\varepsilon \in (0, \min\{\gamma, 1/100, 64\rho^2\gamma^2/9\}]$ and $\delta \in (0,1)$, Algorithm 3 finds a $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 point within $\widetilde{\mathcal{O}}(l\rho^2(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*))/\varepsilon^2)$ gradient evaluations, with probability at least $1-\delta$. The overall time complexity of the algorithm is upper bounded by $\widetilde{\mathcal{O}}(nm^{\omega-1}/\varepsilon^{1.5} + nm/\varepsilon^2)$.*

5 Numerical Experiments

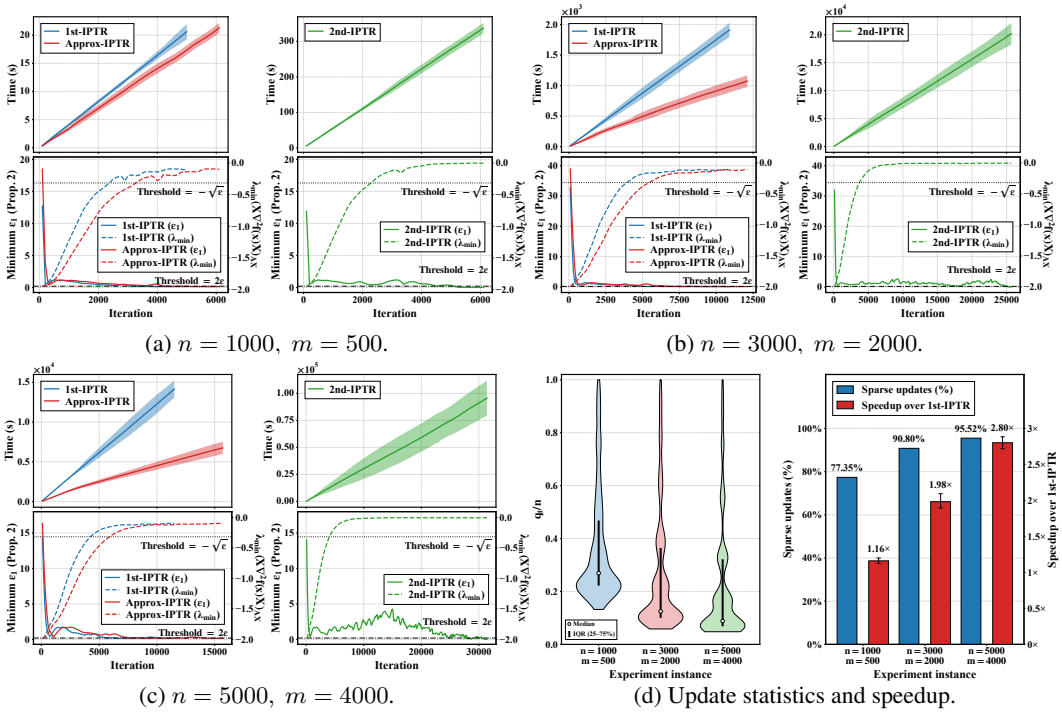
We evaluate our methods from two perspectives: (i) whether our first-order negative-curvature variants converge to approximate second-order KKT points, and (ii) whether the approximate projector improves the practical scalability of first-order IPTR methods.[2] We first consider two nonconvex instances with different optimization landscapes, one with boundary minima and the other with interior minima. We compare Algorithm 1, Algorithm 2, and Algorithm 3 with the exact first- and second-order IPTR methods of Haeser et al. (2019), as shown in Figure 1. All experiments were conducted on an Apple M3 chip with 16 GB memory.

[2]The complete source code for all experiments in Section 5 is available at Anonymous repository.



(a) Boundary minima. (b) Interior minima.

Figure 1: Iteration trajectories on two constrained nonconvex problems with (a) boundary minima and (b) interior minima. For each instance, the left panel compares the first-order IPTR method of Haeser et al. (2019) (1st-IPTR) with our approximate first-order IPTR method (Algorithm 1, Approx-IPTR), while the right panel compares the second-order IPTR method of Haeser et al. (2019) (2nd-IPTR) with our first-order IPTR methods with negative curvature finding, Algorithm 2 (IPTR-NCF) and Algorithm 3 (Approx-IPTR-NCF). In both instances, 1st-IPTR and Approx-IPTR converge to a first-order KKT point, whereas IPTR-NCF and Approx-IPTR-NCF escape this point and converge to a second-order KKT point using only first-order information. Iteration points are plotted every 20 iterations.



(a) $n = 1000,\ m = 500.$ (b) $n = 3000,\ m = 2000.$

(c) $n = 5000,\ m = 4000.$ (d) Update statistics and speedup.

Figure 2: Large-scale empirical evaluation. Panels (a)–(c) correspond to $(n, m) = (1000, 500)$, $(3000, 2000)$, and $(5000, 4000)$, respectively, with same algorithm labels as in Figure 1. For each instance, the upper plots report the running time; solid curves show the mean over five independent runs, and shaded regions indicate $3 \times \mathrm{std}$. The lower plots report convergence to the approximate first- and second-order KKT conditions with $\varepsilon = 0.1$. All quantities are recorded every 100 iterations. In panel (d), left plot shows the distribution of the normalized update size q_t/n, where q_t is the number of coordinates updated at each update step t, while the right plot reports the fraction of iterations that use sparse updates over all iterations and the speedup of Approx-IPTR over 1st-IPTR. Overall, as q_t/n decreases and the fraction of sparse-update iterations increases, the speedup of Approx-IPTR over 1st-IPTR increases.

To evaluate the practical scalability of our algorithms, we consider the quartic problem $\min_{\boldsymbol{x}\geq\mathbf{0},\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}} f(\boldsymbol{x}) := \sum_{i=1}^{n} \frac{\sigma}{4} x_i^4 + \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^\top \boldsymbol{x}$, where $\boldsymbol{Q}$ is chosen such that the objective is nonconvex on $\ker(\boldsymbol{A})$. We evaluate instances with $(n, m) = (1000, 500)$, $(3000, 2000)$, and $(5000, 4000)$ to show that our sparse approximate projection yields increasing computational savings as problem size grows while preserving KKT convergence, as shown in Figure 2.

AI USE STATEMENT

Generative AI tools were used only to aid and polish the writing of this paper. They were not used for any research tasks requiring disclosure under the ICLR 2027 policy. All AI-assisted text was reviewed by the authors, who take responsibility for the final content.

REPRODUCIBILITY STATEMENT

The experimental setup and implementation details are provided in Section 5, and the source code for all experiments is available in the anonymous repository linked therein.

REFERENCES

Zeyuan Allen-Zhu and Yuanzhi Li. Neon2: Finding local minima via first-order oracles. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/d4b2aeb2453bdadaa45cbe9882ffefcf-Paper.pdf. arXiv:1711.06673

Kurt M. Anstreicher. Volumetric path following algorithms for linear programming. *Mathematical Programming*, 76(1):245–263, 1997. https://doi.org/10.1007/BF02614386.

Amir Beck. *First-Order Methods in Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017. doi: 10.1137/1.9781611974997. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611974997. https://epubs.siam.org/doi/abs/10.1137/1.9781611974997.

D. P. Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48 (3):334–334, 1997. doi: 10.1057/palgrave.jors.2600425. https://doi.org/10.1057/palgrave.jors.2600425.

Digvijay Boob, Qi Deng, and Guanghui Lan. Level constrained first order methods for function constrained optimization. *Mathematical Programming*, 209(1):1–61, 2025. https://doi.org/10.1007/s10107-024-02057-4.

Yair Carmon, John C. Duchi, Oliver Hinder, and Aaron Sidford. Accelerated methods for nonconvex optimization. *SIAM Journal on Optimization*, 28(2):1751–1772, 2018. https://doi.org/10.1137/17M1114296.

C. Cartis, N. I. M. Gould, and Ph. L. Toint. An adaptive cubic regularization algorithm for nonconvex optimization with convex constraints and its function-evaluation complexity. *IMA Journal of Numerical Analysis*, 32(4):1662–1695, 2012. doi: 10.1093/imanum/drr035. https://doi.org/10.1093/imanum/drr035.

Coralia Cartis, Nicholas I. M. Gould, and Philippe L. Toint. On the evaluation complexity of cubic regularization methods for potentially rank-deficient nonlinear least-squares problems and its relevance to constrained nonlinear optimization. *SIAM Journal on Optimization*, 23(3):1553–1574, 2013. doi: 10.1137/120869687. URL https://doi.org/10.1137/120869687. https://doi.org/10.1137/120869687.

Coralia Cartis, Nicholas I. M. Gould, and Philippe L. Toint. On the evaluation complexity of constrained nonlinear least-squares and general constrained nonlinear optimization using second-order methods. *SIAM Journal on Numerical Analysis*, 53(2):836–851, 2015. doi: 10.1137/130915546. URL https://doi.org/10.1137/130915546. https://doi.org/10.1137/130915546.

Li Chen, Rasmus Kyng, Yang Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. *Journal of the ACM*, 72 (3):1–103, 2025. https://doi.org/10.1145/3728631.

Michael B. Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. *Journal of the ACM (JACM)*, 68(1):1–39, 2021. https://doi.org/10.1145/3313276.3316303.

James Demmel, Ioana Dumitriu, and Olga Holtz. Fast linear algebra is stable. *Numerische Mathematik*, 108(1):59–91, 2007. https://doi.org/10.1007/s00211-007-0114-x.

Gabriel Haeser, Hongcheng Liu, and Yinyu Ye. Optimality condition and complexity analysis for linearly-constrained optimization without differentiability on the boundary. *Mathematical Programming*, 178(1):263–299, 2019. https://doi.org/10.1007/s10107-018-1290-4.

Chuan He, Zhaosong Lu, and Ting Kei Pong. A Newton-CG based augmented Lagrangian method for finding a second-order stationary point of nonconvex equality constrained optimization with complexity guarantees. *SIAM Journal on Optimization*, 33(3):1734–1766, 2023. doi: 10.1137/22M1489824. URL https://doi.org/10.1137/22M1489824. https://doi.org/10.1137/22M1489824.

Baihe Huang, Shunhua Jiang, Zhao Song, Runzhou Tao, and Ruizhe Zhang. Solving SDP faster: A robust IPM framework and efficient implementation. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 233–244. IEEE, 2022. arXiv:2101.08208

Xinmeng Huang, Shuo Li, Edgar Dobriban, Osbert Bastani, Hamed Hassani, and Dongsheng Ding. One-shot safety alignment for large language models via optimal dualization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 84350–84383. Curran Associates, Inc., 2024. doi: 10.52202/079017-2680. https://proceedings.neurips.cc/paper_files/paper/2024/file/9979a69d2613ab98ad25d3849068f9f0-Paper-Conference.pdf.

Prateek Jain, Chi Jin, Sham Kakade, and Praneeth Netrapalli. Global convergence of non-convex gradient descent for computing matrix squareroot. In *Artificial Intelligence and Statistics*, pp. 479–488. PMLR, 2017. arXiv:1507.05854

Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. Faster dynamic matrix inverse for faster LPs, 2020.

Yuntian Jiang, Chang He, Chuwen Zhang, Dongdong Ge, Bo Jiang, and Yinyu Ye. Beyond nonconvexity: A universal trust-region method with new analyses. *Journal of Scientific Computing*, 106(1):28, 2026. https://doi.org/10.1007/s10915-025-03154-y.

Chi Jin, Praneeth Netrapalli, and Michael I. Jordan. Accelerated gradient descent escapes saddle points faster than gradient descent. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet (eds.), *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pp. 1042–1085. PMLR, 06–09 Jul 2018. URL https://proceedings.mlr.press/v75/jin18a.html. arXiv:1711.10456

N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, STOC '84, pp. 302–311, New York, NY, USA, 1984. Association for Computing Machinery. ISBN 0897911334. doi: 10.1145/800057.808695. URL https://doi.org/10.1145/800057.808695. https://doi.org/10.1145/800057.808695.

Weiwei Kong, Jefferson G. Melo, and Renato D. C. Monteiro. Iteration complexity of an inner accelerated inexact proximal augmented Lagrangian method based on the classical Lagrangian function. *SIAM Journal on Optimization*, 33(1):181–210, 2023. doi: 10.1137/20M136147X. URL https://doi.org/10.1137/20M136147X. https://doi.org/10.1137/20M136147X.

Simon Lacoste-Julien. Convergence rate of Frank–Wolfe for non-convex objectives, 2016. arXiv:1607.00345

Yin Tat Lee and Aaron Sidford. Solving linear programs with $\sqrt{rank}$ linear system solves, 2019.

Yin Tat Lee and Santosh S. Vempala. Tutorial on the robust interior point method, 2021. arXiv:2108.04734

Yin Tat Lee, Zhao Song, and Qiuyi Zhang. Solving empirical risk minimization in the current matrix multiplication time. In *Conference on Learning Theory*, pp. 2140–2157. PMLR, 2019. arXiv:1905.04447

Zichong Li and Yangyang Xu. Augmented Lagrangian–based first-order methods for convex-constrained programs with weakly convex objective. *INFORMS Journal on Optimization*, 3(4): 373–397, 2021. https://doi.org/10.1287/ijoo.2021.0052.

Mingrui Liu, Zhe Li, Xiaoyu Wang, Jinfeng Yi, and Tianbao Yang. Adaptive negative curvature descent with applications in non-convex optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/f52854cc99ae1c1966b0a21d0127975b-Paper.pdf. https://proceedings.neurips.cc/paper_files/paper/2018/file/f52854cc99ae1c1966b0a21d0127975b-Paper.pdf.

Zixuan Liu, Xiaolin Sun, and Zizhan Zheng. Enhancing llm safety via constrained direct preference optimization, 2024. arXiv:2403.02475

Songtao Lu, Meisam Razaviyayn, Bo Yang, Kejun Huang, and Mingyi Hong. Finding second-order stationary points efficiently in smooth nonconvex linearly constrained optimization problems. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 2811–2822. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1da546f25222c1ee710cf7e2f7a3ff0c-Paper.pdf. https://proceedings.neurips.cc/paper_files/paper/2020/file/1da546f25222c1ee710cf7e2f7a3ff0c-Paper.pdf.

Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. *arXiv preprint arXiv:1906.05890*, 2019.

Aryan Mokhtari, Asuman Ozdaglar, and Ali Jadbabaie. Escaping saddle points in constrained optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/069654d5ce089c13f642d19f09a3d1c0-Paper.pdf. arXiv:1809.02162

Michael Muehlebach and Michael I. Jordan. Accelerated first-order optimization under nonlinear constraints. *Mathematical Programming*, pp. 1–46, 2025. https://doi.org/10.1007/s10107-025-02224-1.

Yatin Nandwani, Abhishek Pathak, Mausam, and Parag Singla. A primal dual formulation for deep learning with constraints. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. https://proceedings.neurips.cc/paper_files/paper/2019/file/cf708fc1decf0337aded484f8f4519ae-Paper.pdf.

Yu E. Nesterov and Michael J. Todd. Self-scaled barriers and interior-point methods for convex programming. *Mathematics of Operations Research*, 22(1):1–42, 1997. https://doi.org/10.1287/moor.22.1.1.

Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer, 2006. https://doi.org/10.1007/978-0-387-40065-5.

Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. Cold decoding: Energy-based constrained text generation with langevin dynamics. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 9538–9551. Curran Associates, Inc., 2022. https://proceedings.neurips.cc/paper_files/paper/2022/file/3e25d1aff47964c8409fd5c8dc0438d7-Paper-Conference.pdf.

Alain Rakotomamonjy, Francis Bach, Stéphane Canu, and Yves Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

James Renegar. A polynomial-time algorithm, based on Newton's method, for linear programming. *Mathematical Programming*, 40(1):59–93, 1988. https://doi.org/10.1007/BF01580724.

Ju Sun, Qing Qu, and John Wright. A geometric analysis of phase retrieval. *Foundations of Computational Mathematics*, 18(5):1131–1198, 2018. https://doi.org/10.1007/s10208-017-9365-9.

Pravin M. Vaidya. An algorithm for linear programming which requires $O(((m+n)n^2+(m+n)^{1.5}n)L)$ arithmetic operations. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pp. 29–38, 1987. https://doi.org/10.1145/28395.28399.

Pravin M. Vaidya and David S. Atkinson. A technique for bounding the number of iterations in path following algorithms. In *Complexity in Numerical Optimization*, pp. 462–489. World Scientific, 1993. https://doi.org/10.1142/9789814354363_0021.

Jan van den Brand. A deterministic linear program solver in current matrix multiplication time. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 259–278. SIAM, 2020. https://doi.org/10.1137/1.9781611975994.16.

Jan van den Brand, Yin-Tat Lee, Danupon Nanongkai, Richard Peng, Thatchaphol Saranurak, Aaron Sidford, Zhao Song, and Di Wang. Bipartite matching in nearly-linear time on moderately dense graphs. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 919–930. IEEE, 2020a. arXiv:2009.01802

Jan van den Brand, Yin Tat Lee, Aaron Sidford, and Zhao Song. Solving tall dense linear programs in nearly linear time. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 775–788, 2020b. https://doi.org/10.1145/3357713.3384309.

Jan Van Den Brand, Yang P. Liu, and Aaron Sidford. Dynamic maxflow via dynamic interior point methods. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pp. 1215–1228, 2023. https://doi.org/10.1145/3564246.3585135.

Akifumi Wachi, Thien Q. Tran, Rei Sato, Takumi Tanabe, and Youhei Akimoto. Stepwise alignment for constrained language model policy optimization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 104471–104520. Curran Associates, Inc., 2024. doi: 10.52202/079017-3319. https://proceedings.neurips.cc/paper_files/paper/2024/file/bcfcf7232cb74e1ef82d751880ff835b-Paper-Conference.pdf.

Andreas Wächter and Lorenz T. Biegler. Line search filter methods for nonlinear programming: Motivation and global convergence. *SIAM Journal on Optimization*, 16(1):1–31, 2005. https://doi.org/10.1137/S1052623403426556.

Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006. https://doi.org/10.1007/s10107-004-0559-y.

Guiyun Xiao and Zheng-Jian Bai. A geometric proximal gradient method for sparse least squares regression with probabilistic simplex constraint. *Journal of Scientific Computing*, 92(1):22, 2022.

Shuo Xie and Zhiyuan Li. Implicit bias of adamw: ℓ_∞ norm constrained optimization. *arXiv preprint arXiv:2404.04454*, 2024.

Yi Xu, Rong Jin, and Tianbao Yang. Neon+: Accelerated gradient methods for extracting negative curvature for non-convex optimization, 2017. arXiv:1712.01033

Dongjie Yu, Haitong Ma, Shengbo Li, and Jianyu Chen. Reachability constrained reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 25636–25655. PMLR, 17–23 Jul 2022. https://proceedings.mlr.press/v162/yu22d.html.

Liaoyuan Zeng, Yongle Zhang, Guoyin Li, Ting Kei Pong, and Xiaozhou Wang. Frank–Wolfe-type methods for a class of nonconvex inequality-constrained problems. *Mathematical Programming*, 208(1):717–761, 2024. https://doi.org/10.1007/s10107-023-02055-y.

Chenyi Zhang and Tongyang Li. Escape saddle points by a simple gradient-descent based algorithm. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8545–8556. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/47bd8ac1becf213f155a82244b4a696a-Paper.pdf. arXiv:2111.14069

A Related work

In this section, we review the literature relevant to our work. We begin by discussing prior research on finding approximate KKT points under various conditions, as well as the closely related problem of finding stationary points. Furthermore, given that our primary technical contributions lie in the approximate interior trust-region method and negative curvature finding procedure to find $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 points, we also summarize existing results regarding these techniques.

Approximate KKT points We review several works on computing approximate KKT points for constrained nonconvex optimization problems that are closely related to ours. For problems with linear equality constraints, augmented Lagrangian methods are widely used. Li & Xu (2021) developed a first-order augmented Lagrangian method that attains an 2ε-KKT point in $\mathcal{O}(\varepsilon^{-5/2}\log(1/\varepsilon))$ iterations for problems with a nonconvex objective and convex functional constraints. In a similar vein, Kong et al. (2023) proposed an inner accelerated inexact proximal augmented Lagrangian method for solving linearly constrained smooth nonconvex composite optimization problems, achieving an iteration complexity of $\mathcal{O}(\varepsilon^{-5/2}\log^2(1/\varepsilon))$. These penalty methods generally do not keep the iterates strictly feasible. In the context of interior point methods, Haeser et al. (2019) proposed a first-order and second-order interior-point trust-region algorithm for linearly constrained problems without differentiability on the boundary. They established iteration complexities of $\mathcal{O}(1/\varepsilon^2)$ for finding 2ε-KKT points and $\mathcal{O}(1/\varepsilon^{1.5})$ for $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 points. More recently, Boob et al. (2025) proposed a level-constrained first-order scheme that reduces the problem to a sequence of convex subproblems and achieves an 2ε-KKT point in $\mathcal{O}(\varepsilon^{-2})$ iterations. For KKT points in nonlinear programming, the seminal work by Wächter & Biegler (2006) developed IPOPT, a primal-dual interior point method with filter line-search globalization for large-scale nonlinear programming. Their work provides a practically robust implementation framework, incorporating feasibility restoration, second-order correction, inertia correction, and other practical enhancements. Both their work and ours use the interior point method, and our work adopts a simpler structure with linear equality constraints. We also utilize an approximate update mechanism to improve efficiency. For future research, building upon their theoretical foundation (Wächter & Biegler, 2005) to extend our approach to general nonlinear constraints, and exploring the robust implementation framework presented in their paper, are highly valuable questions.

Approximate stationary points in constrained optimization Distinct from the literature focusing on approximate KKT conditions, a parallel line of research aims to identify FOSP and SOSP for constrained optimization problems. The computation of stationary points has become a standard objective in modern mathematical programming, particularly for large-scale and nonconvex models where guarantees of global optimality are typically unavailable. In the context of FOSP, first-order methods for constrained optimization can be broadly divided into projection-based and projection-free schemes. Projected gradient methods extend gradient descent by projecting each iterate onto the feasible set (Beck, 2017). Projection-free methods such as Frank–Wolfe replace projections with a linear minimization oracle. For possibly nonconvex objectives, Lacoste-Julien (2016) established an $\mathcal{O}(\varepsilon^{-2})$ complexity bound for attaining an ε-FOSP. Recent extensions of this framework include Zeng et al. (2024). Beyond projected gradient and Frank–Wolfe methods, Muehlebach & Jordan (2025) recently proposed a first-order algorithm for nonlinear constrained problems that avoids global optimization over the feasible region while ensuring convergence to an FOSP. When second-order information is accessible, the convergence rate can be improved; for instance, adaptive cubic regularization methods (Cartis et al., 2012; 2013; 2015) are shown to achieve an ε-FOSP within $\mathcal{O}(\varepsilon^{-1.5})$ iterations.

Regarding SOSP, research has focused on leveraging higher-order information to escape saddle points in constrained settings. For problems with generic linear inequality constraints, Lu et al. (2020) analyzed the complexity of finding stationary points using projected gradient-based approaches. They established that first-order methods can reach an $(\varepsilon, \sqrt{\varepsilon})$-SOSP in $\mathcal{O}(\varepsilon^{-2.5})$ iterations, while their second-order counterparts improve this complexity to $\mathcal{O}(\max\{\varepsilon_G^{-2}, \varepsilon_H^{-1.5}\})$ for an $(\varepsilon_G, \varepsilon_H)$-SOSP. For nonconvex equality constriant, a Newton-CG based augmented Lagrangian method proposed by He et al. (2023) can find an $(\varepsilon, \sqrt{\varepsilon})$-SOSP in $\widetilde{O}(\varepsilon^{-7/2})$ iterations. Mokhtari et al. (2018) investigated minimizing smooth nonconvex functions over convex sets, specifically where a ρ-approximate solution to a quadratic program is computable. Their second-order method achieves an (ε, γ)-SOSP with a complexity of $\mathcal{O}(\max\{\varepsilon^{-2}, \rho^{-3}\gamma^{-3}\})$.

Approximate interior point method Since the seminal work by Karmarkar (1984), interior point methods for linear programming have been extensively studied (Anstreicher, 1997; Renegar, 1988; Nesterov & Todd, 1997; Vaidya, 1987; Vaidya & Atkinson, 1993; Lee & Sidford, 2019). In the theoretical computer science literature, a recent line of work studies robust interior point methods. Cohen et al. (2021) established a robust IPM framework where each update only needs to be computed approximately, which is used to prove that linear programming problems can be solved in current matrix multiplication time, whose technique is further derandomized and extended by Jiang et al. (2020), Lee et al. (2019), Lee & Sidford (2019), van den Brand (2020), and van den Brand et al. (2020b). This framework has also been used to develop algorithms for semidefinite programming (Huang et al., 2022; Jiang et al., 2020) and graph problems (Chen et al., 2025; van den Brand et al., 2020a; Van Den Brand et al., 2023). Motivated by this line of work, we follow the same intuition and refer to our method as an approximate interior point method.

B Proof details of Approximate First-Order IPTR Algorithm

B.1 Proof of Lemma 2

In this subsection, we analyze the sparse update scheme for $\overline{\boldsymbol{X}}_t$ and establish the inverse-update complexity in Lemma 2. The key idea is to maintain $\overline{\boldsymbol{x}}_t$ as a sparse approximation of $\boldsymbol{x}_t$ while preserving sufficient accuracy. We first show that the change in $\ln \boldsymbol{x}_t$ between consecutive iterations is small. We then show that $\overline{\boldsymbol{X}}_t$ remains a multiplicative approximation of $\boldsymbol{X}_t$ while changing only a small number of diagonal entries between successive iterations. This sparse update structure allows $(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}$ to be maintained efficiently through low-rank updates, leading to the complexity bound in Lemma 2.

Lemma 3. *Suppose $\boldsymbol{x}_t \in \mathbb{R}^n_{++}$ and $\boldsymbol{x}_{t+1} = \boldsymbol{X}_t(\boldsymbol{e} + \boldsymbol{d}_t)$ with $\|\boldsymbol{d}_t\|_2 \le \beta$ and $0 < \beta < \frac{1}{2}$. Then it holds that $\|\ln \boldsymbol{x}_{t+1} - \ln \boldsymbol{x}_t\|_2 \le 2\beta$.*

Proof. For each i, we have $x_{t+1,i} = x_{t,i}(1 + d_{t,i})$. Taking logarithms gives $\ln x_{t+1,i} - \ln x_{t,i} = \ln(1 + d_{t,i})$. Using the standard inequality for $|d_{t,i}| < \frac{1}{2}$,

$$-2|d_{t,i}| \le -\frac{|d_{t,i}|}{1 - |d_{t,i}|} \le \ln(1 + d_{t,i}) \le |d_{t,i}|, \tag{12}$$

we obtain

$$\|\ln \boldsymbol{x}_{t+1} - \ln \boldsymbol{x}_t\|_2 = \Big(\sum_i \big|\ln(1 + d_{t,i})\big|^2\Big)^{1/2} \le 2\Big(\sum_i |d_{t,i}|^2\Big)^{1/2} = 2\|\boldsymbol{d}_t\|_2 \le 2\beta. \tag{13}$$

□

Lemma 3 shows that the change in $\ln \boldsymbol{x}_t$ across successive iterations is small, which allows us to apply the `SelectVector` algorithm from Lee & Vempala (2021, Algorithm 4). The resulting $\overline{\boldsymbol{x}}_t$ satisfies the two properties in Lemma 1. In particular, Lemma 1(i) gives

$$e^{-\delta}\boldsymbol{x}_t \le \overline{\boldsymbol{x}}_t \le e^{\delta}\boldsymbol{x}_t, \tag{14}$$

so that $\overline{\boldsymbol{X}}_t$ remains a multiplicative approximation of $\boldsymbol{X}_t$, while Lemma 1(ii) bounds the number of diagonal entries updated between consecutive iterations. Using these properties, we next prove Lemma 2.

Denote $\boldsymbol{K}_t := \boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top$ and $\boldsymbol{K}_{t+1} := \boldsymbol{A}\overline{\boldsymbol{X}}_{t+1}^2\boldsymbol{A}^\top = \boldsymbol{A}(\overline{\boldsymbol{X}}_t^2 + \overline{\boldsymbol{X}}_{t+1}^2 - \overline{\boldsymbol{X}}_t^2)\boldsymbol{A}^\top$. Let $\mathcal{I}_t := \{i : \overline{\boldsymbol{x}}_{t+1,i} \neq \overline{\boldsymbol{x}}_{t,i}\}$ be the index set of the updated coordinates, so that $|\mathcal{I}_t| = q_t$. Since only the coordinates in $\mathcal{I}_t$ are updated, the diagonal matrix $\overline{\boldsymbol{X}}_{t+1}^2 - \overline{\boldsymbol{X}}_t^2$ can be written as $\boldsymbol{U}\boldsymbol{C}\boldsymbol{U}^\top$, where $\boldsymbol{U} \in \mathbb{R}^{n\times q_t}$ consists of the columns of the identity matrix $\boldsymbol{I}_n$ indexed by $\mathcal{I}_t$, and $\boldsymbol{C} \in \mathbb{R}^{q_t\times q_t}$ is diagonal. Hence, $\boldsymbol{K}_{t+1} = \boldsymbol{A}(\overline{\boldsymbol{X}}_t^2 + \boldsymbol{U}\boldsymbol{C}\boldsymbol{U}^\top)\boldsymbol{A}^\top = \boldsymbol{K}_t + \boldsymbol{A}\boldsymbol{U}\boldsymbol{C}\boldsymbol{U}^\top\boldsymbol{A}^\top$. The product $\boldsymbol{A}\boldsymbol{U}$ corresponds to a block of $\boldsymbol{A}$ containing m rows and q_t selected columns. We denote this submatrix explicitly as $\boldsymbol{A}_{:,\mathcal{I}_t} := \boldsymbol{A}\boldsymbol{U}$, which is the $m \times q_t$ submatrix of $\boldsymbol{A}$ consisting of the columns indexed by $\mathcal{I}_t$. Applying the Woodbury matrix identity gives

$$\begin{aligned}\boldsymbol{K}_{t+1}^{-1} &= \left(\boldsymbol{K}_t + \boldsymbol{A}_{:,\mathcal{I}_t}\boldsymbol{C}\boldsymbol{A}_{:,\mathcal{I}_t}^\top\right)^{-1} \\ &= \boldsymbol{K}_t^{-1} - \boldsymbol{K}_t^{-1}\boldsymbol{A}_{:,\mathcal{I}_t}\left(\boldsymbol{C}^{-1} + \boldsymbol{A}_{:,\mathcal{I}_t}^\top\boldsymbol{K}_t^{-1}\boldsymbol{A}_{:,\mathcal{I}_t}\right)^{-1}\boldsymbol{A}_{:,\mathcal{I}_t}^\top\boldsymbol{K}_t^{-1}.\end{aligned} \tag{15}$$

When $q_t \leq m$, computing $\left(\boldsymbol{C}^{-1} + \boldsymbol{A}_{:,\mathcal{I}_t}^\top\boldsymbol{K}_t^{-1}\boldsymbol{A}_{:,\mathcal{I}_t}\right)^{-1}$ requires $\mathcal{O}(m^2q_t^{\omega-2} + mq_t^{\omega-1} + q_t^\omega) = \mathcal{O}(m^2q_t^{\omega-2})$ time, which accounts for one matrix multiplication of size $q_t \times m$ with $m \times m$, one multiplication of size $q_t \times m$ with $m \times q_t$, and one inversion of size $(q_t \times q_t)$. The time complexity of multiplying an $m \times n$ matrix with an $n \times q$ matrix is $\mathcal{O}(mnq\min\{m,n,q\}^{\omega-3})$ because the rectangular matrix multiplication can be decomposed into blocks of square submatrices along its smallest dimension. Computing $\boldsymbol{A}_{:,\mathcal{I}_t}^\top\boldsymbol{K}_t^{-1}$ and $\boldsymbol{K}_t^{-1}\boldsymbol{A}_{:,\mathcal{I}_t}$ also requires $\mathcal{O}(m^2q_t^{\omega-2})$ time. Finally, multiplying $\boldsymbol{K}_t^{-1}\boldsymbol{A}_{:,\mathcal{I}_t}$, $\left(\boldsymbol{C}^{-1} + \boldsymbol{A}_{:,\mathcal{I}_t}^\top\boldsymbol{K}_t^{-1}\boldsymbol{A}_{:,\mathcal{I}_t}\right)^{-1}$, and $\boldsymbol{A}_{:,\mathcal{I}_t}^\top\boldsymbol{K}_t^{-1}$ together has a time complexity of $\mathcal{O}(mq_t^{\omega-1} + m^2q_t^{\omega-2}) = \mathcal{O}(m^2q_t^{\omega-2})$, due to one multiplication of size $m \times q_t$ with $q_t \times q_t$ and one multiplication of size $m \times q_t$ with $q_t \times m$. In summary, when $q_t \leq m$, the overall time complexity for computing the inverse $\boldsymbol{K}_{t+1}^{-1}$ is $\mathcal{O}(m^2q_t^{\omega-2})$.

Now consider the case $m < q_t \leq n$. In this case, computing

$$\left(\boldsymbol{C}^{-1} + \boldsymbol{A}_{:,\mathcal{I}_t}^\top\boldsymbol{K}_t^{-1}\boldsymbol{A}_{:,\mathcal{I}_t}\right)^{-1}$$

requires $\mathcal{O}(q_t^\omega)$ time, as it involves the inversion of a $q_t \times q_t$ matrix. As this time complexity may exceed the cost of recomputing the inverse from scratch, we instead directly form $\boldsymbol{K}_{t+1} = \boldsymbol{A}\overline{\boldsymbol{X}}_{t+1}^2\boldsymbol{A}^\top$ and compute its inverse anew. Forming $\boldsymbol{K}_{t+1}$ requires $\mathcal{O}(nm^{\omega-1})$ time, which accounts for one matrix multiplication of size $m \times n$ with $n \times m$, and computing the inverse of $\boldsymbol{K}_{t+1}$ requires $\mathcal{O}(m^\omega)$ time. Therefore, in this case, the overall time complexity is $\mathcal{O}(nm^{\omega-1})$.

B.2 Proof of Proposition 3

The total computational cost of Algorithm 1 arises from three main components: (1) updating the inverse in the projection matrix $\boldsymbol{R}_t$, (2) computing the approximate direction $\widetilde{\boldsymbol{d}}_t$ and updating the iterates $\boldsymbol{x}_{t+1}$, and (3) executing the `SelectVector` procedure that determines the sparse update pattern of $\overline{\boldsymbol{X}}_t$. We analyze each component in turn.

1. Cost of updating inverse $(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}$: According to Lemma 1, the maximum number of coordinate changes between $\ln\overline{\boldsymbol{x}}_t$ and $\ln\overline{\boldsymbol{x}}_{t-1}$ is bounded by $\mathcal{O}(4^{l_t}\log^2 n)$. Let q_t denote the actual number of changed coordinates at iteration t. Then, there exists a constant $C \geq 1$ such that
$$q_t \leq \min\left\{n, C4^{l_t}\log^2 n\right\}, \text{ for } t = 1, 2, \ldots, T. \tag{16}$$
We categorize q_t according to the value of l_t. Recall that l_t is defined as the largest integer $l \leq \lceil\log n\rceil$ such that $t \equiv 0 \bmod 2^l$. Let N_l denote the number of indices t that share the same value of l, i.e.,
$$\begin{aligned}N_l &:= |\{t \in [1,T],\ t \equiv 0 \bmod 2^l \text{ and } t \not\equiv 0 \bmod 2^{l+1}\}| \text{ for } 0 \leq l \leq \lceil\log n\rceil - 1, \\ N_{\lceil\log n\rceil} &:= |\{t \in [1,T],\ t \equiv 0 \bmod 2^{\lceil\log n\rceil}\}|.\end{aligned} \tag{17}$$
For $0 \leq l \leq \lceil\log n\rceil - 1$, the size of N_l equals the number of integers in $[1,T]$ that are divisible by 2^l but not by 2^{l+1}
$$N_l = \left\lfloor\frac{T}{2^l}\right\rfloor - \left\lfloor\frac{T}{2^{l+1}}\right\rfloor. \tag{18}$$

In addition, $N_{\lceil \log n \rceil} = \left\lfloor \frac{T}{2^{\lceil \log n \rceil}} \right\rfloor$. Combining these bounds, we obtain $N_l \le \frac{T}{2^l}$ for $0 \le l \le \lceil \log n \rceil$.

At each iteration t, if q_t coordinates are updated, then by Lemma 2, when $q_t \le m$, the inverse $(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}$ can be updated from the previous inverse via a rank-q_t Woodbury update in time $\mathcal{O}(m^2 q_t^{\omega-2})$, where ω denotes the exponent of matrix multiplication. When $m < q_t \le n$, we compute $(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}$ directly rather than applying the Woodbury update. The time complexity of this direct computation is $\mathcal{O}(nm^{\omega-1})$, which accounts for forming $\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top$ and computing its inverse. Therefore, we apply the Woodbury update only when $q_t \le m$, and compute the inverse directly otherwise. We set l^* to ensure that for $l = 0, 1, \ldots, l^*$, $C4^l \log^2 n \le m$.

$$l^* := \frac{1}{2}\left\lfloor \left(\log_2 \left(\frac{m}{C \log^2 n} \right) \right) \right\rfloor. \tag{19}$$

For $q_t \le m$, the time complexity is

$$\begin{aligned} \sum_{t=1}^{T} m^2 q_t^{\omega-2} &\le \sum_{l=0}^{\lceil \log n \rceil} m^2 N_l \left(C4^l \log^2 n\right)^{\omega-2} \\ &\le C^{\omega-2} m^2 T \log^{2(\omega-2)} n \sum_{l=0}^{\lceil \log n \rceil} 2^{(2\omega-5)l} \\ &= \widetilde{\mathcal{O}}\left(m^2 T\right). \end{aligned} \tag{20}$$

For $q_t > m$, which implies $l_t > l^*$ and $T > \sqrt{\frac{m}{C \log^2 n}}$ (otherwise we would have $N_{l_t} = 0$), the time complexity is

$$\begin{aligned} \sum_{t=1}^{T} 1_{\{q_t > m\}} nm^{\omega-1} &\le \sum_{t=1}^{T} 1_{\{C4^{l_t} \log^2 n > m\}} nm^{\omega-1} \\ &= \sum_{l=l^*+1}^{\lceil \log n \rceil} N_l nm^{\omega-1} \\ &\le Tnm^{\omega-1} \sum_{l=l^*+1}^{\lceil \log n \rceil} \frac{1}{2^l} \\ &\le Tnm^{\omega-1} \left(\frac{1}{2^{l^*}} \right) \\ &\le Tnm^{\omega-1} \frac{\sqrt{C} \log n}{\sqrt{m}} \\ &= \widetilde{\mathcal{O}}\left(nm^{\omega-1.5} T\right). \end{aligned} \tag{21}$$

2. Cost of computing $\widetilde{\boldsymbol{d}}_t$ and updating $\boldsymbol{x}_{t+1}$. After computing the inverse $(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}$, we obtain $\widetilde{\boldsymbol{d}}_t$ according to (8). This step requires a time complexity of $\mathcal{O}(mn)$ per iteration due to the matrix–vector multiplication. Updating $\boldsymbol{x}_{t+1}$ incurs an additional cost of $\mathcal{O}(n)$ per iteration. Therefore, the overall time complexity of this part over T iterations is $\mathcal{O}(mnT)$.
3. Cost of `SelectVector` algorithm: For each iteration $t \in [1, T]$, the `SelectVector` algorithm in Lee & Vempala (2021, Algorithm 4) examines every level $l \in [0, \lceil \log_2 n \rceil]$ satisfying $t = 0 \bmod 2^l$, and compares each coordinate $i \in [n]$ between $\ln \boldsymbol{x}_{t,i}$ and $\ln \boldsymbol{x}_{(t-2^l),i}$. This process requires a total computational cost of

$$\sum_{l=1}^{\lceil \log_2 n \rceil} nlN_l \le nT \sum_{l=1}^{\lceil \log_2 n \rceil} \frac{l}{2^l} \le 2nT, \tag{22}$$

where N_l is defined in (17). Therefore, the time complexity of this part is $\mathcal{O}(nT)$.

Combining the above bounds and including the initial cost $\widetilde{\mathcal{O}}(nm^{\omega-1})$ of forming $(\boldsymbol{A}\overline{\boldsymbol{X}}_0^2\boldsymbol{A}^\top)^{-1}$ gives the claimed complexity bound $\widetilde{\mathcal{O}}(nm^{\omega-1}+mnT)$.

B.3 Proof of Proposition 4

To prove Proposition 4, we first quantify the error introduced by replacing the exact orthogonal projector $\boldsymbol{P}_t$ with the approximate projector $\boldsymbol{R}_t$. The following lemma shows that the multiplicative approximation between $\overline{\boldsymbol{X}}_t$ and $\boldsymbol{X}_t$ yields a corresponding spectral-norm bound between the two projectors, while preserving feasibility of the projected directions.

Lemma 4. *Let $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ be a full-row-rank matrix, and let $\boldsymbol{x}\in\mathbb{R}^n_{++}$ satisfy $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$. Denote $\boldsymbol{X}:=\mathrm{diag}(\boldsymbol{x})$. Suppose there exists another positive diagonal matrix $\overline{\boldsymbol{X}}\in\mathbb{R}^{n\times n}_{++}$ that approximates $\boldsymbol{X}$ such that, for some $0<\delta<\frac{1}{2}$, $e^{-\delta}\boldsymbol{X}\le\overline{\boldsymbol{X}}\le e^{\delta}\boldsymbol{X}$. Define the orthogonal projector onto $\ker(\boldsymbol{A}\boldsymbol{X})$ as*

$$\boldsymbol{P}=\boldsymbol{I}-\boldsymbol{X}\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{X}^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}, \tag{23}$$

and define

$$\boldsymbol{R}=\boldsymbol{I}-\boldsymbol{X}^{-1}\overline{\boldsymbol{X}}^2\boldsymbol{A}^\top(\boldsymbol{A}\overline{\boldsymbol{X}}^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}. \tag{24}$$

Then $\boldsymbol{R}$ is a projection matrix satisfying $\boldsymbol{A}\boldsymbol{X}\boldsymbol{R}\boldsymbol{h}=\boldsymbol{0}$ for any $\boldsymbol{h}\in\mathbb{R}^n$, and $\|\boldsymbol{R}-\boldsymbol{P}\|\le 46\delta$.

Proof. For clarity, let us introduce the shorthand notation $\boldsymbol{M}:=\boldsymbol{X}^{-1}\overline{\boldsymbol{X}}^2\boldsymbol{X}^{-1}$, $\boldsymbol{B}:=\boldsymbol{A}\boldsymbol{X}$. Under this notation, we can rewrite $\boldsymbol{R}=\boldsymbol{I}-\boldsymbol{M}\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}$, and $\boldsymbol{P}=\boldsymbol{I}-\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}$.

First, we show that $\boldsymbol{R}$ is a projection. It suffices to verify that $\boldsymbol{R}$ is idempotent, i.e., $\boldsymbol{R}^2=\boldsymbol{R}$. By direct calculation,

$$\begin{aligned}\boldsymbol{R}^2&=\left(\boldsymbol{I}-\boldsymbol{M}\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\right)^2\\&=\boldsymbol{I}-2\boldsymbol{M}\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}+\boldsymbol{M}\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\,\boldsymbol{M}\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\\&=\boldsymbol{I}-\boldsymbol{M}\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\\&=\boldsymbol{R}.\end{aligned} \tag{25}$$

Thus, $\boldsymbol{R}$ is indeed a projection matrix.

Next, we show that the range of $\boldsymbol{R}$ is contained in $\ker(\boldsymbol{B})$. To see this, note that

$$\boldsymbol{B}\boldsymbol{R}=\boldsymbol{B}-\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}=\boldsymbol{B}-\boldsymbol{B}=\boldsymbol{0}. \tag{26}$$

Hence, for any $\boldsymbol{h}$, we have $\boldsymbol{B}\boldsymbol{R}\boldsymbol{h}=\boldsymbol{0}$, and consequently $\boldsymbol{A}\boldsymbol{X}\boldsymbol{R}\boldsymbol{h}=\boldsymbol{0}$.

Having established the basic algebraic properties of $\boldsymbol{R}$, we now proceed to bound the spectral norm difference between $\boldsymbol{R}$ and $\boldsymbol{P}$. By expanding their definitions and applying the triangle inequality along with the sub-multiplicativity of the spectral norm, we obtain

$$\begin{aligned}\|\boldsymbol{R}-\boldsymbol{P}\|&=\|(\boldsymbol{I}-\boldsymbol{M})\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}+\boldsymbol{M}\left(\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}-\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\right)\|\\&\le\|(\boldsymbol{I}-\boldsymbol{M})\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\|+\|\boldsymbol{M}\left(\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}-\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\right)\|\\&\le\|\boldsymbol{I}-\boldsymbol{M}\|\|\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\|+\|\boldsymbol{M}\|\|\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}(\boldsymbol{M}-\boldsymbol{I})\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\|\\&\le\|\boldsymbol{I}-\boldsymbol{M}\|+\|\boldsymbol{M}\|\|\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\|\|\boldsymbol{I}-\boldsymbol{M}\|\|\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\|\\&=\|\boldsymbol{I}-\boldsymbol{M}\|+\|\boldsymbol{M}\|\|\boldsymbol{I}-\boldsymbol{M}\|\|\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\|.\end{aligned} \tag{27}$$

where the last equality holds because $\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}$ is an orthogonal projection matrix, meaning its spectral norm is exactly 1.

To evaluate the remaining terms in this bound, we observe that since $\boldsymbol{M}$ is a positive diagonal matrix, its spectral norm equals its maximum eigenvalue, i.e., $\|\boldsymbol{M}\|=\lambda_{\max}(M)$. It then remains to bound the norm $\|\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\|$. Noting that $\boldsymbol{M}\succeq\lambda_{\min}(\boldsymbol{M})\boldsymbol{I}$, it follows that

$$\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top\succeq\lambda_{\min}(\boldsymbol{M})\boldsymbol{B}\boldsymbol{B}^\top. \tag{28}$$

Taking the inverse reverses the Loewner order:

$$(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1} \preceq \frac{1}{\lambda_{\min}(\boldsymbol{M})}(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}. \tag{29}$$

Consequently,

$$\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B} \preceq \frac{1}{\lambda_{\min}(\boldsymbol{M})}\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B} \preceq \frac{1}{\lambda_{\min}(\boldsymbol{M})}\boldsymbol{I}, \tag{30}$$

where we again used the fact that $\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}$ is an orthogonal projector whose eigenvalues are at most 1. This implies

$$\|\boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{M}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\| \le \frac{1}{\lambda_{\min}(\boldsymbol{M})}. \tag{31}$$

Substituting this bound back into our norm inequality (27) yields

$$\|\boldsymbol{R}-\boldsymbol{P}\| \le \|\boldsymbol{I}-\boldsymbol{M}\|(1+\kappa(\boldsymbol{M})), \tag{32}$$

where $\kappa(\boldsymbol{M}) := \lambda_{\max}(\boldsymbol{M})/\lambda_{\min}(\boldsymbol{M})$ denotes the condition number of $\boldsymbol{M}$.

Finally, we express this bound in terms of the approximation error δ. By the assumption that $e^{-\delta}\boldsymbol{X} \le \overline{\boldsymbol{X}} \le e^{\delta}\boldsymbol{X}$, we equivalently have $e^{-2\delta}\boldsymbol{I} \le \boldsymbol{M} \le e^{2\delta}\boldsymbol{I}$. This relation immediately implies that $\kappa(\boldsymbol{M}) \le e^{4\delta}$, and the spectral norm $\|\boldsymbol{I}-\boldsymbol{M}\|$ is bounded by $\max\{e^{2\delta}-1, 1-e^{-2\delta}\} = e^{2\delta}-1$. Using the elementary inequality $e^x - 1 \le xe^x$ for $x > 0$, we can further bound $\|\boldsymbol{I}-\boldsymbol{M}\| \le 2\delta e^{\delta}$. Substituting these bounds into (32) gives

$$\|\boldsymbol{R}-\boldsymbol{P}\| \le \|\boldsymbol{I}-\boldsymbol{M}\|(1+\kappa(\boldsymbol{M})) \le 2\delta e^{2\delta}(1+e^{4\delta}). \tag{33}$$

Plugging in $\delta < \frac{1}{2}$ bounds the constant factor by $2e(1+e^2) \le 46$, yielding $\|\boldsymbol{R}-\boldsymbol{P}\| \le 46\delta$ and completing the proof. □

With the approximation error between $\boldsymbol{R}_t$ and $\boldsymbol{P}_t$ controlled, we now turn to the one-step analysis of Proposition 4. We compare the approximate direction $\widetilde{\boldsymbol{d}}_t$ with the exact trust-region solution $\boldsymbol{d}_t$ and show that each iteration either produces sufficient decrease in the potential function or satisfies the desired approximate KKT conditions.

Let $\boldsymbol{d}_t$ denote the exact optimal solution of (5). By the necessary and sufficient optimality conditions, there exist $\lambda_t \ge 0$ and $\boldsymbol{v}_t \in \mathbb{R}^m$ such that

$$\boldsymbol{X}_t\nabla f(\boldsymbol{x}_t) - \varepsilon\boldsymbol{e} + \boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t + \lambda_t\boldsymbol{d}_t = 0. \tag{34}$$

We now examine the decrease of the potential function when the update $\boldsymbol{x}_{t+1} := \boldsymbol{X}_t(\boldsymbol{e}+\widetilde{\boldsymbol{d}}_t)$ is applied:

$$\begin{aligned}\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) &\le \langle \boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t), \widetilde{\boldsymbol{d}}_t\rangle + \frac{l}{2}\|\widetilde{\boldsymbol{d}}_t\|^2 + \varepsilon\beta^2 \\ &= \langle -\boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t - \lambda_t\boldsymbol{d}_t, \widetilde{\boldsymbol{d}}_t\rangle + \left(\frac{l}{2}+\varepsilon\right)\beta^2 \\ &= -\lambda_t\|\boldsymbol{d}_t\|^2 + \lambda_t\boldsymbol{d}_t^\top(\boldsymbol{d}_t - \widetilde{\boldsymbol{d}}_t) + \left(\frac{l}{2}+\varepsilon\right)\beta^2.\end{aligned} \tag{35}$$

The second equality follows from (34), and the third is due to $\boldsymbol{A}\boldsymbol{X}_t\widetilde{\boldsymbol{d}}_t = \boldsymbol{0}$. This expresses the potential decrease in terms of the exact direction $\boldsymbol{d}_t$ and its approximation $\widetilde{\boldsymbol{d}}_t$. We evaluate this bound under two cases: (i) $\lambda_t = 0$ and $\|\boldsymbol{d}_t\| < \beta$, or (ii) $\lambda_t > 0$ and $\|\boldsymbol{d}_t\| = \beta$.

Case 1. $\lambda_t = 0$ and $\|\boldsymbol{d}_t\| < \beta$. In this case, (34) yields $\boldsymbol{X}_t(\nabla f(\boldsymbol{x}_t) + \boldsymbol{A}^\top\boldsymbol{v}_t) = \varepsilon\boldsymbol{e}$. It then follows that $\|\boldsymbol{X}_t(\nabla f(\boldsymbol{x}_t)+\boldsymbol{A}^\top\boldsymbol{v}_t)\|_\infty = \varepsilon < 2\varepsilon$. Furthermore, since the diagonal matrix is strictly positive, this equality directly implies $\nabla f(\boldsymbol{x}_t) + \boldsymbol{A}^\top\boldsymbol{v}_t > 0$.

Case 2. Under the conditions $\lambda_t > 0$ and $\|\boldsymbol{d}_t\| = \beta$, we distinguish between two subcases according to the norm of the projected gradient $\|\widetilde{\boldsymbol{d}}_t\|$.

Case 2.1. If $\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)=0$, then, by the definition of $\boldsymbol{R}_t$,

$$\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)=\left(\boldsymbol{X}_t^{-1}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}_t\right)\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t).$$

Let $\boldsymbol{M}_t:=\boldsymbol{X}_t^{-1}\overline{\boldsymbol{X}}_t^2\boldsymbol{X}_t^{-1}$ and $\boldsymbol{v}_t:=(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}_t^2\nabla\phi(\boldsymbol{x}_t)$. Then

$$\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)=\boldsymbol{M}_t\boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t,$$

and hence

$$\begin{aligned}\|\boldsymbol{X}_t\nabla f(\boldsymbol{x}_t)-\varepsilon\boldsymbol{e}+\boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t\|_\infty&=\|(\boldsymbol{I}-\boldsymbol{M}_t)\boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t\|_\infty\\&\le\|(\boldsymbol{I}-\boldsymbol{M}_t)\boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t\|\\&\le\|\boldsymbol{I}-\boldsymbol{M}_t\|\|\boldsymbol{X}_t\boldsymbol{A}^\top(\boldsymbol{A}\overline{\boldsymbol{X}}_t^2\boldsymbol{A}^\top)^{-1}\boldsymbol{A}\boldsymbol{X}_t\|\|\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|\\&\le 2\delta e^{2\delta}\cdot e^{2\delta}\cdot L_\phi\\&\le\varepsilon.\end{aligned}\tag{36}$$

The fourth line follows from (31) in Lemma 4, and the last line follows from $\delta\le 1/2$ and $\delta\le\varepsilon/(15L_\phi)$. Thus, there exists $\boldsymbol{v}_t$ such that $\|\boldsymbol{X}_t\nabla f(\boldsymbol{x}_t)-\varepsilon\boldsymbol{e}+\boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t\|_\infty\le\varepsilon$. This implies that $0\le\boldsymbol{X}_t(\nabla f(\boldsymbol{x}_t)+\boldsymbol{A}^\top\boldsymbol{v}_t)\le 2\varepsilon$ holds elementwise, and $\nabla f(\boldsymbol{x}_t)+\boldsymbol{A}^\top\boldsymbol{v}_t\ge 0$ since $\boldsymbol{X}_t$ is nonnegative.

Case 2.2. When $\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\neq 0$, to bound the decrease in the potential function, it remains to control the error term $\lambda_t\boldsymbol{d}_t^\top(\boldsymbol{d}_t-\widetilde{\boldsymbol{d}}_t)$ in (35). By the definition of $\widetilde{\boldsymbol{d}}_t$, we have $\|\widetilde{\boldsymbol{d}}_t\|=\beta$. Therefore,

$$\begin{aligned}\|\lambda_t\boldsymbol{d}_t^\top(\boldsymbol{d}_t-\widetilde{\boldsymbol{d}}_t)\|&\le\lambda_t\beta^2\left\|\frac{\boldsymbol{P}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)}{\|\boldsymbol{P}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|}-\frac{\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)}{\|\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|}\right\|\\&\le 2\beta\|\boldsymbol{P}_t-\boldsymbol{R}_t\|\|\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|\\&\le 2\beta\cdot 46\delta\cdot L_\phi\le\beta^2.\end{aligned}\tag{37}$$

The second line follows from the unit-vector difference bound $\|\boldsymbol{a}/\|\boldsymbol{a}\|-\boldsymbol{b}/\|\boldsymbol{b}\|\|\le 2\|\boldsymbol{a}-\boldsymbol{b}\|/\|\boldsymbol{a}\|$ and the identity $\|\boldsymbol{P}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|=\|\boldsymbol{P}_t\boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t+\lambda_t\boldsymbol{P}_t\boldsymbol{d}_t\|=\lambda_t\|\boldsymbol{d}_t\|$, while the last line follows from Lemma 4 and $\delta\le\beta/(92L_\phi)$.

To simplify the notation in the subsequent analysis, define $p(\boldsymbol{x}_t,\boldsymbol{v}_t):=\boldsymbol{X}_t\nabla f(\boldsymbol{x}_t)-\varepsilon\boldsymbol{e}+\boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t$. From (34), $\|p(\boldsymbol{x}_t,\boldsymbol{v}_t)\|=\lambda_t\beta$. Then the potential decrease satisfies

$$\phi(\boldsymbol{x}_{t+1})-\phi(\boldsymbol{x}_t)\le-\lambda_t\beta^2+\left(\frac{l}{2}+\varepsilon+1\right)\beta^2=-\|p(\boldsymbol{x}_t,\boldsymbol{v}_t)\|\beta+\left(\frac{l}{2}+\varepsilon+1\right)\beta^2.\tag{38}$$

Case 2.2.1. If $\|p(\boldsymbol{x}_t,\boldsymbol{v}_t)\|\ge\varepsilon$, then

$$\phi(\boldsymbol{x}_{t+1})-\phi(\boldsymbol{x}_t)\le-\varepsilon\beta+\left(\frac{l}{2}+\varepsilon+1\right)\beta^2\le-\frac{\varepsilon^2}{2l+4\varepsilon+4}.$$

Case 2.2.2. If $\|p(\boldsymbol{x}_t,\boldsymbol{v}_t)\|<\varepsilon$, then by the definition of $p(\boldsymbol{x}_t,\boldsymbol{v}_t)$, we have $\|\boldsymbol{X}_t\nabla f(\boldsymbol{x}_t)-\varepsilon\boldsymbol{e}+\boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t\|_\infty\le\varepsilon$. This again implies that $0\le\boldsymbol{X}_t(\nabla f(\boldsymbol{x}_t)+\boldsymbol{A}^\top\boldsymbol{v}_t)\le 2\varepsilon$ holds elementwise, and $\nabla f(\boldsymbol{x}_t)+\boldsymbol{A}^\top\boldsymbol{v}_t\ge 0$ since $\boldsymbol{X}_t$ is nonnegative.

B.4 PROOF OF THEOREM 1

By Proposition 4, under the choice of parameters $\beta=\varepsilon/(l+2\varepsilon+2)$ and $\delta=\min\{\varepsilon/(15L_\phi),\beta/(92L_\phi)\}$, each iteration of Algorithm 1 either ensures a sufficient decrease in the potential function by at least $\varepsilon^2/(2l+4\varepsilon+4)$ or indicates that the current iterate $\boldsymbol{x}_t$ is already a 2ε-KKT point.

Suppose first that every iteration falls into the decrease case. Since the initial point $\boldsymbol{x}_0>\boldsymbol{0}$ is assumed to satisfy $-\sum_{i=1}^n\ln(x_i)\ge-\sum_{i=1}^n\ln(x_{0,i})-C_0$ for all $\boldsymbol{x}\in\Omega^\circ$, after t iterations we obtain

$$f(\boldsymbol{x}_t)-f(\boldsymbol{x}_0)=\phi(\boldsymbol{x}_t)+\varepsilon\sum_{i=1}^n\ln(x_{t,i})-\phi(\boldsymbol{x}_0)+\varepsilon\sum_{i=1}^n\ln(x_{0,i})\le-\frac{t\varepsilon^2}{2l+4\varepsilon+4}+\varepsilon C_0.\tag{39}$$

Consequently, the total number of iterations is bounded by $T = \frac{(f(\boldsymbol{x}_0)-f(\boldsymbol{x}^*)+(C_0-1)\varepsilon)(2l+4\varepsilon+4)}{\varepsilon^2}$, where $f(\boldsymbol{x}^*)$ denotes the global minimum value. If the algorithm reaches this maximum number of iterations, it must hold that $f(\boldsymbol{x}_t) - f(\boldsymbol{x}^*) \le \varepsilon$.

Finally, by Proposition 3, the overall time complexity under the above iteration bound is $\widetilde{\mathcal{O}}(nm^{\omega-1}+nm/\varepsilon^2)$, which completes the proof.

Remark 1. *In practice, the trust-region radius β in Algorithm 1 can be adjusted adaptively as in Jiang et al. (2026). Indeed, the proof of Proposition 4 shows that the decrease of the potential function satisfies*

$$\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) \le -\|p(\boldsymbol{x}_t, \boldsymbol{v}_t)\|\beta + \left(\frac{l}{2} + \varepsilon + 1\right)\beta^2.$$

Here, $\|p(\boldsymbol{x}_t, \boldsymbol{v}_t)\| = \|\boldsymbol{P}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|$, which is close to $\|\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|$ under the approximate projection. This suggests that, in practical implementations, one may choose β on the order of $\beta = \mathcal{O}(\|\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\|/l)$. Such a choice allows a larger step and potentially faster decrease when the projected gradient is large. For a small projected gradient, β is correspondingly small, which reduces the potential decrease. Therefore, the current stopping criterion still applies.

B.5 Proof of Corollary 1

We now consider the special case where f is concave on Ω°, i.e.,

$$f(\boldsymbol{y}) \le f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^\top(\boldsymbol{y} - \boldsymbol{x}), \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \Omega^\circ.$$

In this setting, f admits a first-order upper bound along every feasible scaled direction. The potential decrease can be bounded directly at order ε, improving the iteration complexity of first-order IPTR algorithms from $\mathcal{O}(1/\varepsilon^2)$ to $\mathcal{O}(1/\varepsilon)$. We establish this result for both the exact-projection and approximate-projection implementations.

We first consider the exact-projection implementation. At each iteration, we solve (5) with trust-region radius β, compute the search direction according to (6), and update $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{X}_t\boldsymbol{d}_t$. The resulting iteration and time complexity bounds are stated in the following proposition.

Proposition 7. *Suppose that f is concave on Ω°. Consider the exact first-order IPTR algorithm described above, with trust-region radius $\beta \in (0, 1)$. Then, for any $\varepsilon \in (0, 1]$, the algorithm either returns a 2ε-KKT point within $\mathcal{O}(1/\varepsilon)$ iterations, or returns an iterate $\boldsymbol{x}_t$ such that $f(\boldsymbol{x}_t) - f(\boldsymbol{x}^*) \le \varepsilon$. The overall time complexity is upper bounded by $\mathcal{O}(nm^{\omega-1}/\varepsilon)$.*

Proof. Let $\boldsymbol{d}_t$ be defined by (6), and set $\boldsymbol{x}_{t+1} = \boldsymbol{X}_t(\boldsymbol{e} + \boldsymbol{d}_t)$. By the necessary and sufficient optimality conditions for (5), there exist $\lambda_t \ge 0$ and $\boldsymbol{v}_t \in \mathbb{R}^m$ such that (34) holds. Repeating the derivation of (35) with $\widetilde{\boldsymbol{d}}_t = \boldsymbol{d}_t$, and using the concavity of f, we obtain

$$\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) \le -\lambda_t\|\boldsymbol{d}_t\|^2 + \varepsilon\beta^2. \tag{40}$$

We distinguish two cases. If $\lambda_t = 0$ and $\|\boldsymbol{d}_t\| < \beta$, then as in Case 1 of Proposition 4, there exists $\boldsymbol{v}_t \in \mathbb{R}^m$ such that $\boldsymbol{X}_t(\nabla f(\boldsymbol{x}_t) + \boldsymbol{A}^\top\boldsymbol{v}_t) = \varepsilon\boldsymbol{e}$. Thus $\boldsymbol{x}_t$ is a 2ε-KKT point. Otherwise, we have $\lambda_t > 0$ and $\|\boldsymbol{d}_t\| = \beta$. Define $p(\boldsymbol{x}_t, \boldsymbol{v}_t) := \boldsymbol{X}_t\nabla f(\boldsymbol{x}_t) - \varepsilon\boldsymbol{e} + \boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t$. By (34), we have $\|p(\boldsymbol{x}_t, \boldsymbol{v}_t)\| = \lambda_t\|\boldsymbol{d}_t\|$. If $\|p(\boldsymbol{x}_t, \boldsymbol{v}_t)\| < \varepsilon$, then as in Case 2.2.2 of Proposition 4, the same 2ε-KKT conclusion follows. Otherwise,

$$\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) \le -\lambda_t\|\boldsymbol{d}_t\|^2 + \varepsilon\beta^2 \le -\varepsilon\beta + \varepsilon\beta^2 = -\varepsilon\beta(1-\beta). \tag{41}$$

Therefore, at each iteration, either $\boldsymbol{x}_t$ is a 2ε-KKT point, or the potential function decreases by at least $\varepsilon\beta(1-\beta)$. Combining this with the initialization of $\boldsymbol{x}_0$, after t iterations we have

$$f(\boldsymbol{x}_t) - f(\boldsymbol{x}_0) = \phi(\boldsymbol{x}_t) + \varepsilon\sum_{i=1}^n \ln(x_{t,i}) - \phi(\boldsymbol{x}_0) + \varepsilon\sum_{i=1}^n \ln(x_{0,i}) \le -t\varepsilon\beta(1-\beta) + \varepsilon C_0. \tag{42}$$

It follows that the total number of iterations is bounded by $T = \frac{f(\boldsymbol{x}_0)-f(\boldsymbol{x}^*)+(C_0-1)\varepsilon}{\varepsilon\beta(1-\beta)}$, where $f(\boldsymbol{x}^*)$ denotes the global minimum value. If the algorithm reaches this bound without returning a 2ε-KKT point, then it must hold that $f(\boldsymbol{x}_t) - f(\boldsymbol{x}^*) \le \varepsilon$. The time complexity of each iteration is $\mathcal{O}(nm^{\omega-1})$ as discussed after (6). Multiplying by the iteration bound yields the overall time complexity $\mathcal{O}(nm^{\omega-1}/\varepsilon)$. □

We next analyze the time and iteration complexity of Algorithm 1 for concave functions. The descent analysis closely follows the previous arguments and still gives the $\mathcal{O}(1/\varepsilon)$ iteration complexity. The approximate update scheme differs in one respect. In the concave setting, the trust-region radius satisfies $\beta = \Theta(1)$. To maintain the accuracy of the approximate projection $\boldsymbol{R}_t$, Lemma 4 gives $\|\boldsymbol{R}_t - \boldsymbol{P}_t\| \le 46\delta$ under $e^{-\delta}\boldsymbol{X}_t \le \overline{\boldsymbol{X}_t} \le e^{\delta}\boldsymbol{X}_t$. Since the projection error now needs to be controlled at the $\Theta(\varepsilon)$ level, we take $\delta = \Theta(\varepsilon)$. With a constant stepsize but an update tolerance of order $\Theta(\varepsilon)$, the approximate update scheme needs to update more coordinates as shown in Lemma 1. We first establish the corresponding one-step guarantee for the Algorithm 1.

Lemma 5. *Suppose that f is concave on Ω° and Assumption 3 holds. For any $\varepsilon \in (0,1]$, $\beta \in (0,1)$, choose $\delta \le (1-\beta)\varepsilon/184L_\phi$. At iteration t, suppose that $e^{-\delta}\boldsymbol{x}_t \le \overline{\boldsymbol{x}}_t \le e^{\delta}\boldsymbol{x}_t$, define $\widetilde{\boldsymbol{d}}_t$ by (8), and let $\boldsymbol{x}_{t+1} = \boldsymbol{X}_t(\boldsymbol{e} + \widetilde{\boldsymbol{d}}_t)$. Then exactly one of the following two statements holds:*

$$\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) \le -\frac{1}{2}\varepsilon\beta(1-\beta), \tag{43}$$

or there exists $\boldsymbol{v}_t \in \mathbb{R}^m$ such that

$$\|\boldsymbol{X}_t(\nabla f(\boldsymbol{x}_t) + \boldsymbol{A}^\top \boldsymbol{v}_t)\|_\infty \le 2\varepsilon, \qquad \nabla f(\boldsymbol{x}_t) + \boldsymbol{A}^\top \boldsymbol{v}_t \ge \boldsymbol{0}. \tag{44}$$

Proof. Let $\boldsymbol{d}_t$ denote the exact optimal solution of (5). By the necessary and sufficient optimality conditions for (5), there exist $\lambda_t \ge 0$ and $\boldsymbol{v}_t \in \mathbb{R}^m$ such that (34) holds. Repeating the derivation of (35), with the concavity of f, we obtain

$$\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) \le -\lambda_t\|\boldsymbol{d}_t\|^2 + \lambda_t \boldsymbol{d}_t^\top(\boldsymbol{d}_t - \widetilde{\boldsymbol{d}}_t) + \varepsilon\beta^2. \tag{45}$$

If $\lambda_t = 0$ and $\|\boldsymbol{d}_t\| < \beta$, then, as in Case 1 of Proposition 4, we can find $\boldsymbol{v}_t \in \mathbb{R}^m$ such that $\boldsymbol{X}_t(\nabla f(\boldsymbol{x}_t) + \boldsymbol{A}^\top \boldsymbol{v}_t) = \varepsilon \boldsymbol{e}$. Thus $\boldsymbol{x}_t$ is a 2ε-KKT point.

If $\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t) = \boldsymbol{0}$, then, by the same argument as in Case 2.1 of Proposition 4, one can choose $\boldsymbol{v}_t$ such that

$$\|\boldsymbol{X}_t\nabla f(\boldsymbol{x}_t) - \varepsilon\boldsymbol{e} + \boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t\|_\infty \le 2\delta e^{4\delta}L_\phi.$$

Since $\delta \le (1-\beta)\varepsilon/184L_\phi$ and $\beta \in (0,1)$, it follows that $2\delta e^{4\delta}L_\phi \le \varepsilon$. Thus $\boldsymbol{x}_t$ is again a 2ε-KKT point.

Now suppose that $\boldsymbol{R}_t\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t) \ne \boldsymbol{0}$ and $\lambda_t > 0$, $\|\boldsymbol{d}_t\| = \beta$. By the definition of $\widetilde{\boldsymbol{d}}_t$, we have $\|\widetilde{\boldsymbol{d}}_t\| = \beta$. As in Case 2.2 of Proposition 4,

$$|\lambda_t \boldsymbol{d}_t^\top(\boldsymbol{d}_t - \widetilde{\boldsymbol{d}}_t)| \le 2\beta\|\boldsymbol{P}_t - \boldsymbol{R}_t\|\,\|\boldsymbol{X}_t\nabla\phi(\boldsymbol{x}_t)\| \le 92\beta\delta L_\phi \le \frac{1}{2}\varepsilon\beta(1-\beta). \tag{46}$$

Define $p(\boldsymbol{x}_t, \boldsymbol{v}_t) := \boldsymbol{X}_t\nabla f(\boldsymbol{x}_t) - \varepsilon\boldsymbol{e} + \boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t$. By (34), we have $\|p(\boldsymbol{x}_t, \boldsymbol{v}_t)\| = \lambda_t\beta$. If $\|p(\boldsymbol{x}_t, \boldsymbol{v}_t)\| < \varepsilon$, then, by the same argument as in Case 2.2.2 of Proposition 4, the first-order KKT condition holds. Otherwise, $\lambda_t\beta \ge \varepsilon$, and hence

$$\begin{aligned}
\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) &\le -\lambda_t\|\boldsymbol{d}_t\|^2 + \varepsilon\beta^2 + \frac{1}{2}\varepsilon\beta(1-\beta) \\
&\le -\varepsilon\beta + \varepsilon\beta^2 + \frac{1}{2}\varepsilon\beta(1-\beta) \\
&= -\frac{1}{2}\varepsilon\beta(1-\beta).
\end{aligned} \tag{47}$$

□

Based on Lemma 5, we obtain the iteration and time complexity bounds for the approximate-projection implementation.

Proposition 8. *Suppose f is concave on Ω° and Assumption 3 holds. Let $\varepsilon \in (0,1]$. Consider Algorithm 1 with the trust-region radius $\beta \in (0,1)$ and $\delta = (1-\beta)\varepsilon/(184L_\phi)$. Then, the algorithm terminates within $\mathcal{O}(1/\varepsilon)$ iterations, returning either a 2ε-KKT point or an iterate $\boldsymbol{x}_t$ satisfying $f(\boldsymbol{x}_t) - f(\boldsymbol{x}^*) \le \varepsilon$. Furthermore, the overall time complexity is $\widetilde{\mathcal{O}}(nm^{\omega-1} + nm/\varepsilon^2)$.*

Proof. According to Lemma 5, each iteration either yields a 2ε-KKT point or decreases the potential function by at least $\frac{1}{2}\varepsilon\beta(1-\beta)$. Following the same potential argument used in Proposition 7, the total number of iterations T is bounded by

$$T = \frac{2(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + (C_0 - 1)\varepsilon)}{\varepsilon\beta(1-\beta)}. \tag{48}$$

To analyze the time complexity, let q_t be the number of coordinates updated between $\ln \overline{\boldsymbol{x}}_{t-1}$ and $\ln \overline{\boldsymbol{x}}_t$. By Lemma 1(ii), there exists a constant $C \geq 1$ such that

$$q_t \leq \min\left\{n,\ C4^{l_t}(\beta/\delta)^2 \log^2 n\right\}, \qquad t = 1, 2, \ldots, T. \tag{49}$$

Substituting $\delta = (1-\beta)\varepsilon/(184L_\phi)$ yields $(\beta/\delta)^2 = (184\beta L_\phi)^2/((1-\beta)^2\varepsilon^2)$. This allows us to rewrite equation 49 as

$$q_t \leq \min\left\{n,\ C'4^{l_t} \log^2 n/\varepsilon^2\right\}, \qquad t = 1, 2, \ldots, T, \tag{50}$$

where $C' = C(184\beta L_\phi/(1-\beta))^2$.

Let N_l denote the number of iterations sharing the same value of l, as defined in (17). Computationally, the Woodbury matrix identity is more efficient when $q_t \leq m$, whereas direct projection is preferable when $m < q_t \leq n$. To formalize this transition, we define a threshold l_ε^* such that $C'4^l \log^2 n/\varepsilon^2 \leq m$ holds for all $l \leq l_\varepsilon^*$. Specifically, we set

$$l_\varepsilon^* := \frac{1}{2}\left\lfloor \log_2\left(\frac{m\varepsilon^2}{C' \log^2 n}\right)\right\rfloor. \tag{51}$$

For iterations where $q_t \leq m$ (and thus $l_t \leq l_\varepsilon^*$), the cumulative time complexity is bounded by

$$\begin{aligned}
\sum_{t:\, l_t \leq l_\varepsilon^*} m^2 q_t^{\omega-2} &\leq \sum_{l=0}^{l_\varepsilon^*} m^2 N_l \left(\frac{C'4^l \log^2 n}{\varepsilon^2}\right)^{\omega-2} \\
&\leq \frac{C'^{\omega-2} m^2 T \log^{2(\omega-2)} n}{\varepsilon^{2\omega-4}} \sum_{l=0}^{l_\varepsilon^*} 2^{(2\omega-5)l} \\
&= \widetilde{\mathcal{O}}\left(\frac{m^2}{\varepsilon^{2\omega-3}}\right).
\end{aligned} \tag{52}$$

For iterations where $q_t > m$, the total cost is

$$\begin{aligned}
\sum_{t:\, l_t > l_\varepsilon^*} nm^{\omega-1} &= \sum_{l=l_\varepsilon^*+1}^{\lceil \log n \rceil} N_l\, nm^{\omega-1} \\
&\leq Tnm^{\omega-1} \sum_{l=l_\varepsilon^*+1}^{\lceil \log n \rceil} \frac{1}{2^l} \\
&\leq Tnm^{\omega-1} \frac{1}{2^{l_\varepsilon^*}} \\
&\leq \widetilde{\mathcal{O}}\left(\frac{nm^{\omega-1}T}{\varepsilon\sqrt{m}}\right) \\
&= \widetilde{\mathcal{O}}\left(\frac{nm^{\omega-1.5}}{\varepsilon^2}\right).
\end{aligned} \tag{53}$$

Beyond the matrix updates, computing $\widetilde{\boldsymbol{d}}_t$ and updating $\boldsymbol{x}_{t+1}$ requires $\mathcal{O}(mn)$ operations per iteration, contributing $\mathcal{O}(mn/\varepsilon)$ across all T iterations. Factoring in the $\widetilde{\mathcal{O}}(nm^{\omega-1})$ initial cost to construct $(\boldsymbol{A}\overline{\boldsymbol{X}}_0^2\boldsymbol{A}^\top)^{-1}$, the aggregate time complexity becomes

$$\widetilde{\mathcal{O}}\left(nm^{\omega-1} + \frac{mn}{\varepsilon} + \frac{m^2}{\varepsilon^{2\omega-3}} + \frac{nm^{\omega-1.5}}{\varepsilon^2}\right). \tag{54}$$

Since $m \leq n$ and $\omega < 2.5$, the above bound reduces to $\widetilde{\mathcal{O}}(nm^{\omega-1} + nm/\varepsilon^2)$. □

Remark 2. *In fact, as noted above, the sparse-update analysis applies only to iterations with $q_t \le m$, equivalently, to those satisfying $l_t \le l_\varepsilon^*$. Hence, when $l_\varepsilon^* < 0$, this regime is empty, and the overall complexity bound reduces to the direct per-iteration cost $\mathcal{O}(nm^{\omega-1})$. Therefore, in the concave case, Algorithm 1 admits the overall time complexity $\min\{\widetilde{\mathcal{O}}(nm^{\omega-1} + nm/\varepsilon^2), \mathcal{O}(nm^{\omega-1}/\varepsilon)\}$. Up to logarithmic factors, the comparison is determined by nm/ε^2 and $nm^{\omega-1}/\varepsilon$. Hence, the bound $\widetilde{\mathcal{O}}(nm^{\omega-1} + nm/\varepsilon^2)$ is better when $\varepsilon = \Omega(m^{2-\omega})$, whereas the bound $\mathcal{O}(nm^{\omega-1}/\varepsilon)$ is better when $\varepsilon = O(m^{2-\omega})$. Thus, the approximate update implementation is preferable when moderate accuracy suffices, whereas the direct per-iteration implementation is preferable when higher accuracy is required.*

Together, Proposition 7 and Proposition 8 establish Corollary 1.

C Proof of Finding Second-Order KKT Points by First-Order Methods

C.1 Proof of Proposition 5

We first show that $\boldsymbol{y}_t \in \ker(\boldsymbol{AX})$ for all $t \ge 0$. We proceed by induction. The base case holds since $\boldsymbol{y}_0 \in \ker(\boldsymbol{AX})$ by initialization. Assume $\boldsymbol{y}_t \in \ker(\boldsymbol{AX})$. The update rule gives:

$$\boldsymbol{AX}\boldsymbol{y}_{t+1} = \boldsymbol{AX}\boldsymbol{y}_t - \frac{\|\boldsymbol{y}_t\|}{lr}\boldsymbol{AXP}\left(\nabla f\left(\boldsymbol{X}\left(\boldsymbol{e} + \frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|}\right)\right) - \nabla f(\boldsymbol{x})\right).$$

Since $\boldsymbol{P}$ is the orthogonal projection onto $\ker(\boldsymbol{AX})$, we have $\boldsymbol{AXP} = \boldsymbol{0}$. Thus, $\boldsymbol{AX}\boldsymbol{y}_{t+1} = \boldsymbol{AX}\boldsymbol{y}_t = \boldsymbol{0}$, meaning $\boldsymbol{y}_{t+1} \in \ker(\boldsymbol{AX})$. Moreover, as long as $r < 1$, the point $\boldsymbol{x}' := \boldsymbol{X}\left(\boldsymbol{e} + r\boldsymbol{y}_t/\|\boldsymbol{y}_t\|\right)$ satisfies $x_i' = x_i\left(1 + ry_{t,i}/\|\boldsymbol{y}_t\|\right) > 0$. We also have $\boldsymbol{A}\boldsymbol{x}' = \boldsymbol{Ax} + \boldsymbol{AX}r\boldsymbol{y}_t/\|\boldsymbol{y}_t\| = \boldsymbol{b}$. Therefore, $\boldsymbol{x}' \in \Omega^\circ$.

Let us define the approximation error Δ as

$$\Delta := \frac{\|\boldsymbol{y}_t\|}{lr}\boldsymbol{PX}\left(\nabla f(\boldsymbol{X}(\boldsymbol{e} + \frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|})) - \nabla f(\boldsymbol{x}) - \nabla^2 f(\boldsymbol{x})\boldsymbol{X}\frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|}\right). \tag{55}$$

With this definition, the update rule can be rewritten as

$$\boldsymbol{y}_{t+1} = \boldsymbol{y}_t - (\Delta + \frac{1}{l}\boldsymbol{PX}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\boldsymbol{y}_t) = \left(I - \frac{1}{l}\boldsymbol{PX}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\right)\boldsymbol{y}_t - \Delta. \tag{56}$$

By Taylor's theorem, there exists $\xi \in [0, 1]$ such that the norm of Δ can be bounded using Assumption 2

$$\begin{aligned}
\frac{\|\Delta\|}{\|\boldsymbol{y}_t\|} &= \frac{1}{lr}\left\|\boldsymbol{PX}\left(\nabla f(\boldsymbol{X}(\boldsymbol{e} + \frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|})) - \nabla f(\boldsymbol{x}) - \nabla^2 f(\boldsymbol{x})\boldsymbol{X}\frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|}\right)\right\| \\
&\le \frac{1}{lr}\left\|\boldsymbol{X}\left(\nabla^2 f(\boldsymbol{X}(\boldsymbol{e} + \frac{\xi r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|}))\frac{r\boldsymbol{X}\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|} - \nabla^2 f(\boldsymbol{x})\boldsymbol{X}\frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|}\right)\right\| \\
&\le \frac{1}{l}\left\|\left(\boldsymbol{X}\nabla^2 f(\boldsymbol{X}(\boldsymbol{e} + \frac{\xi r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|}))\boldsymbol{X} - \boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\right)\right\| \\
&\le \frac{\rho}{l}\left\|\frac{\xi r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|}\right\| \\
&\le \frac{\rho r}{l}.
\end{aligned} \tag{57}$$

The eigenvectors of the projected scaled Hessian $\boldsymbol{PX}\nabla^2 f(\boldsymbol{x})\boldsymbol{XP}$ restricted to $\ker(\boldsymbol{AX})$ form an orthogonal basis. Let $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_k$ be its eigenvalues, with corresponding eigenvectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k$. By assumption, there exist indices p and p' such that

$$\lambda_p \le -\sqrt{\varepsilon} \le \lambda_{p+1}, \quad \text{and} \quad \lambda_{p'} \le -\sqrt{\varepsilon}/2 < \lambda_{p'+1}. \tag{58}$$

We define the subspaces $\mathcal{S}_\| = \operatorname{span}\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_p\}$ and $\mathcal{S}_\perp = \operatorname{span}\{\boldsymbol{u}_{p+1}, \ldots, \boldsymbol{u}_k\}$. Similarly, we define $\mathcal{S}_{\|'} = \operatorname{span}\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{p'}\}$ and $\mathcal{S}_{\perp'} = \operatorname{span}\{\boldsymbol{u}_{p'+1}, \ldots, \boldsymbol{u}_k\}$.

Before proceeding, we establish the properties of the initial vector $\boldsymbol{y}^0$. Since $\boldsymbol{y}^0$ is initialized uniformly at random from the unit sphere in the k-dimensional subspace $\ker(\boldsymbol{AX})$, standard results on random projections guarantee that its projection onto the subspace $\mathcal{S}_{\|}$ (which has dimension $p \geq 1$) satisfies

$$\mathbb{P}\left(\frac{\|\boldsymbol{y}_{0,\|}\|}{\|\boldsymbol{y}_0\|} \geq \sqrt{\frac{\pi}{k}}\delta_0\right) \geq 1 - \delta_0. \tag{59}$$

We condition on this high-probability event for the remainder of the proof and define $\alpha_0 := \frac{\|\boldsymbol{y}_{0,\|}\|}{\|\boldsymbol{y}_0\|} \geq \sqrt{\frac{\pi}{k}}\delta_0$.

Projecting $\boldsymbol{y}_{t+1}$ onto the subspace $\mathcal{S}_{\|}$ yields

$$\begin{aligned}
\|\boldsymbol{y}_{t+1,\|}\| &\geq \left(1 + \frac{\sqrt{\varepsilon}}{l}\right)\|\boldsymbol{y}_{t,\|}\| - \|\Delta\| \\
&\geq \left(1 + \frac{\sqrt{\varepsilon}}{l} - \frac{\|\Delta\|}{\|\boldsymbol{y}_t\|}\frac{\|\boldsymbol{y}_t\|}{\|\boldsymbol{y}_{t,\|}\|}\right)\|\boldsymbol{y}_{t,\|}\| \\
&\geq \left(1 + \frac{\sqrt{\varepsilon}}{l} - \frac{\rho r}{l}\frac{\|\boldsymbol{y}_t\|}{\|\boldsymbol{y}_{t,\|}\|}\right)\|\boldsymbol{y}_{t,\|}\|.
\end{aligned} \tag{60}$$

Let $\alpha_t := \frac{\|\boldsymbol{y}_{t,\|}\|}{\|\boldsymbol{y}_t\|}$. The parameter r is chosen such that $r \leq \frac{\alpha_0}{8\rho}\sqrt{\varepsilon}\left(\frac{1+7\sqrt{\varepsilon}/(8l)}{2}\right)^{\mathcal{T}}$, which ensures $\frac{\rho r}{l\alpha_0\left(\frac{1+7\sqrt{\varepsilon}/(8l)}{2}\right)^{\mathcal{T}}} \leq \frac{\sqrt{\varepsilon}}{8l}$. We will prove by induction that $\alpha_t \geq \alpha_0\left(\frac{1+7\sqrt{\varepsilon}/(8l)}{2}\right)^t$. Suppose this holds for step t, which implies $\frac{\rho r}{l\alpha_t} \leq \frac{\sqrt{\varepsilon}}{8l}$. Then at step $t+1$,

$$\begin{aligned}
\|\boldsymbol{y}_{t+1,\|}\| &\geq \left(1 + \frac{\sqrt{\varepsilon}}{l} - \frac{\rho r}{l}\frac{\|\boldsymbol{y}_t\|}{\|\boldsymbol{y}_{t,\|}\|}\right)\|\boldsymbol{y}_{t,\|}\|, \\
\frac{\|\boldsymbol{y}_{t+1,\|}\|}{\|\boldsymbol{y}_{t+1}\|} &\geq \frac{\|\boldsymbol{y}_{t,\|}\|}{\|\boldsymbol{y}_t\|}\frac{\|\boldsymbol{y}_t\|}{\|\boldsymbol{y}_{t+1}\|}\left(1 + \frac{\sqrt{\varepsilon}}{l} - \frac{\rho r}{l}\frac{\|\boldsymbol{y}_t\|}{\|\boldsymbol{y}_{t,\|}\|}\right), \\
\alpha_{t+1} &\geq \alpha_t\frac{\|\boldsymbol{y}_t\|}{\|\boldsymbol{y}_{t+1}\|}\left(1 + \frac{\sqrt{\varepsilon}}{l} - \frac{\rho r}{l\alpha_t}\right).
\end{aligned} \tag{61}$$

Next, we establish an lower bound on the ratio $\|\boldsymbol{y}_t\|/\|\boldsymbol{y}_{t+1}\|$. Using the triangle inequality and Assumption 1, we have

$$\begin{aligned}
\|\boldsymbol{y}_{t+1}\| &\leq \|\boldsymbol{y}_t\|\left(1 + \frac{1}{lr}\left\|\boldsymbol{P}\boldsymbol{X}\left(\nabla f(\boldsymbol{X}(\boldsymbol{e} + \frac{r\boldsymbol{y}_t}{\|\boldsymbol{y}_t\|})) - \nabla f(\boldsymbol{x})\right)\right\|\right) \\
&\leq \|\boldsymbol{y}_t\|\left(1 + \frac{1}{lr}\cdot lr\right) \\
&\leq 2\|\boldsymbol{y}_t\|.
\end{aligned} \tag{62}$$

Combining equation 61 and equation 62, we obtain

$$\alpha_{t+1} \geq \alpha_t\frac{1 + \frac{\sqrt{\varepsilon}}{l} - \frac{\rho r}{l\alpha_t}}{2} \geq \alpha_t\frac{1 + \frac{7\sqrt{\varepsilon}}{8l}}{2} \geq \alpha_0\left(\frac{1 + 7\sqrt{\varepsilon}/(8l)}{2}\right)^{t+1}. \tag{63}$$

This completes the induction. Consequently, for all $1 \leq t \leq \mathcal{T}$, we have $\frac{\rho r}{l\alpha_t} \leq \frac{\sqrt{\varepsilon}}{8l}$ and

$$\|\boldsymbol{y}_{t,\|}\| \geq \left(1 + \frac{7\sqrt{\varepsilon}}{8l}\right)\|\boldsymbol{y}_{t-1,\|}\| \geq \left(1 + \frac{7\sqrt{\varepsilon}}{8l}\right)^t\|\boldsymbol{y}_{0,\|}\|. \tag{64}$$

Let $\boldsymbol{y}_{t,\perp'}$ denote the projection of $\boldsymbol{y}_t$ onto the subspace $\mathcal{S}_{\perp'}$. We will show that there exists $1 \leq t_0 \leq \mathcal{T}$ such that $\|\boldsymbol{y}_{t_0,\perp'}\|/\|\boldsymbol{y}_{t_0}\| \leq \frac{\sqrt{\varepsilon}}{8l}$. Suppose, for the sake of contradiction, that $\|\boldsymbol{y}_{t,\perp'}\|/\|\boldsymbol{y}_t\| > \frac{\sqrt{\varepsilon}}{8l}$

holds for all $1 \le t \le \mathcal{T}$. Under this assumption, we have

$$\begin{aligned}
\|\boldsymbol{y}_{t+1,\perp'}\| &\le \left(1+\frac{\sqrt{\varepsilon}}{2l}\right)\|\boldsymbol{y}_{t,\perp'}\| + \|\Delta\| \\
&\le \left(1+\frac{\sqrt{\varepsilon}}{2l}+\frac{\|\Delta\|}{\|\boldsymbol{y}_t\|}\frac{\|\boldsymbol{y}_t\|}{\|\boldsymbol{y}_{t,\perp'}\|}\right)\|\boldsymbol{y}_{t,\perp'}\| \\
&\le \left(1+\frac{\sqrt{\varepsilon}}{2l}+\frac{\rho r}{l}\frac{\|\boldsymbol{y}_t\|}{\|\boldsymbol{y}_{t,\perp'}\|}\right)\|\boldsymbol{y}_{t,\perp'}\| \\
&\le \left(1+\frac{\sqrt{\varepsilon}}{2l}+\frac{\rho r}{l}\frac{8l}{\sqrt{\varepsilon}}\right)\|\boldsymbol{y}_{t,\perp'}\| \\
&\le \left(1+\frac{5\sqrt{\varepsilon}}{8l}\right)\|\boldsymbol{y}_{t,\perp'}\|.
\end{aligned} \tag{65}$$

Combining this with the initial condition $\mathcal{P}\left(\|\boldsymbol{y}_{\|}^0\|/\|\boldsymbol{y}^0\| \ge \sqrt{\frac{\pi}{k}}\delta_0\right) \ge 1-\delta_0$ and $\mathcal{T} = \frac{8l}{\sqrt{\varepsilon}}\log\left(\frac{8l}{\delta_0}\sqrt{\frac{n}{\pi\varepsilon}}\right)$, we obtain

$$\frac{\|\boldsymbol{y}_{\mathcal{T},\perp'}\|}{\|\boldsymbol{y}_{\mathcal{T},\|}\|} \le \frac{\|\boldsymbol{y}_{0,\perp'}\|\left(1+\frac{5\sqrt{\varepsilon}}{8l}\right)^{\mathcal{T}}}{\|\boldsymbol{y}_{0,\|}\|\left(1+\frac{7\sqrt{\varepsilon}}{8l}\right)^{\mathcal{T}}} \le \frac{1}{\delta_0}\sqrt{\frac{k}{\pi}}\left(\frac{1+\frac{5\sqrt{\varepsilon}}{8l}}{1+\frac{7\sqrt{\varepsilon}}{8l}}\right)^{\mathcal{T}} \le \frac{1}{\delta_0}\sqrt{\frac{n}{\pi}}\left(\frac{1+\frac{5\sqrt{\varepsilon}}{8l}}{1+\frac{7\sqrt{\varepsilon}}{8l}}\right)^{\mathcal{T}} \le \frac{\sqrt{\varepsilon}}{8l}. \tag{66}$$

This contradicts the assumption that $\|\boldsymbol{y}_{t,\perp'}\|/\|\boldsymbol{y}_t\| > \frac{\sqrt{\varepsilon}}{8l}$ for all $1 \le t \le \mathcal{T}$. Therefore, there exists $1 \le t_0 \le \mathcal{T}$ such that $\|\boldsymbol{y}_{t_0,\perp'}\|/\|\boldsymbol{y}_{t_0}\| \le \frac{\sqrt{\varepsilon}}{8l}$. Let $\widehat{\boldsymbol{e}} := \boldsymbol{y}_{t_0}/\|\boldsymbol{y}_{t_0}\|$ denotes the normalized vector. Then $\boldsymbol{A}\boldsymbol{X}\widehat{\boldsymbol{e}} = \boldsymbol{0}$, $\|\widehat{\boldsymbol{e}}_{\perp'}\| \le \frac{\sqrt{\varepsilon}}{8l}$, and $\|\widehat{\boldsymbol{e}}_{\|'}\|^2 \ge 1-\frac{\varepsilon}{64l^2}$. Consequently,

$$\widehat{\boldsymbol{e}}^\top\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\widehat{\boldsymbol{e}} = \widehat{\boldsymbol{e}}_{\perp'}^\top\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\widehat{\boldsymbol{e}}_{\perp'} + \widehat{\boldsymbol{e}}_{\|'}^\top\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\widehat{\boldsymbol{e}}_{\|'} \le l\|\widehat{\boldsymbol{e}}_{\perp'}\|^2 - \sqrt{\varepsilon}\|\widehat{\boldsymbol{e}}_{\|'}\|^2/2 \le -\sqrt{\varepsilon}/4. \tag{67}$$

This completes the proof of the proposition.

C.2 Proof of Proposition 6

Since $\widehat{\boldsymbol{e}}$ is a unit vector and $\varepsilon \le \min\{1/100, 64\rho^2\gamma^2/9\}$, the scaled step has norm $3\sqrt{\varepsilon}/(8\rho) \le \min\{\gamma, 3/80\} < 1$. Together with $\boldsymbol{A}\boldsymbol{X}\widehat{\boldsymbol{e}} = \boldsymbol{0}$, this ensures that the updated point remains in Ω°. Combining Lemma 9 with the logarithmic-barrier bound in Appendix D, we obtain

$$\begin{aligned}
&\phi\left(\boldsymbol{X}\left(\boldsymbol{e} - \frac{\langle\nabla\phi(\boldsymbol{x}), \boldsymbol{X}\widehat{\boldsymbol{e}}\rangle}{|\langle\nabla\phi(\boldsymbol{x}), \boldsymbol{X}\widehat{\boldsymbol{e}}\rangle|}\cdot\frac{3\sqrt{\varepsilon}}{8\rho}\widehat{\boldsymbol{e}}\right)\right) \\
&\le \phi(\boldsymbol{x}) + \left\langle\boldsymbol{X}\nabla\phi(\boldsymbol{x}), -\frac{\langle\nabla\phi(\boldsymbol{x}), \boldsymbol{X}\widehat{\boldsymbol{e}}\rangle}{|\langle\nabla\phi(\boldsymbol{x}), \boldsymbol{X}\widehat{\boldsymbol{e}}\rangle|}\cdot\frac{3\sqrt{\varepsilon}}{8\rho}\widehat{\boldsymbol{e}}\right\rangle \\
&\quad + \frac{1}{2}\cdot\frac{9\varepsilon}{64\rho^2}\widehat{\boldsymbol{e}}^\top\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\widehat{\boldsymbol{e}} + \frac{\rho}{6}\cdot\frac{27\sqrt{\varepsilon^3}}{512\rho^3} + \frac{9\varepsilon^2}{128\rho^2\left(1-\frac{3\sqrt{\varepsilon}}{8\rho}\right)} \\
&\le \phi(\boldsymbol{x}) + \frac{9\varepsilon}{128\rho^2}\widehat{\boldsymbol{e}}^\top\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\widehat{\boldsymbol{e}} + \frac{9\sqrt{\varepsilon^3}}{1024\rho^2} + \frac{9\varepsilon^2}{128\rho^2\left(1-\frac{3\sqrt{\varepsilon}}{8\rho}\right)} \\
&\le \phi(\boldsymbol{x}) - \frac{9\sqrt{\varepsilon^3}}{512\rho^2} + \frac{9\sqrt{\varepsilon^3}}{1024\rho^2} + \frac{9\varepsilon^2}{128\rho^2\left(1-\frac{3\sqrt{\varepsilon}}{8\rho}\right)} \\
&\le \phi(\boldsymbol{x}) - \frac{9\sqrt{\varepsilon^3}}{512\rho^2} + \frac{9\sqrt{\varepsilon^3}}{1024\rho^2} + \frac{\sqrt{\varepsilon^3}}{128\rho^2} \\
&= \phi(\boldsymbol{x}) - \frac{\sqrt{\varepsilon^3}}{1024\rho^2},
\end{aligned} \tag{68}$$

where the first inequality uses $\boldsymbol{X}\nabla\phi(\boldsymbol{x}) = \boldsymbol{X}\nabla f(\boldsymbol{x}) - \varepsilon\boldsymbol{e}$ to combine the linear terms of the objective and the logarithmic barrier. The second inequality follows from the choice of sign, which makes the linear term nonpositive, and the third inequality follows from the negative curvature condition. For the fourth inequality, $\sqrt{\varepsilon} \le 1/10$ give

$$\frac{9\varepsilon^2}{128\rho^2\left(1-\frac{3\sqrt{\varepsilon}}{8\rho}\right)} = \frac{\sqrt{\varepsilon^3}}{128\rho^2}\frac{9\sqrt{\varepsilon}}{1-\frac{3\sqrt{\varepsilon}}{8\rho}} \le \frac{\sqrt{\varepsilon^3}}{128\rho^2}\cdot\frac{9/10}{1-3/80} = \frac{72}{77}\cdot\frac{\sqrt{\varepsilon^3}}{128\rho^2} \le \frac{\sqrt{\varepsilon^3}}{128\rho^2}. \tag{69}$$

This completes the proof.

C.3 Proof of Theorem 2

We now prove Theorem 2 by combining the first-order descent guarantee with the negative-curvature results established above. The following lemma characterizes each first-order IPTR iteration: either the potential function decreases sufficiently, or the current iterate already satisfies the 2ε-KKT conditions. We then distinguish the latter case according to whether the second-order KKT condition is satisfied.

Lemma 6 (Haeser et al. (2019, Theorem 2)). *Under Assumption 1 and Assumption 2, For any $\varepsilon \in (0, \min\{\gamma, 1\}]$, either the following condition holds at iteration t:*

$$\phi(\boldsymbol{x}_{t+1}) - \phi(\boldsymbol{x}_t) \le -\frac{\varepsilon^2}{4l+4\varepsilon} \tag{70}$$

or $\|\boldsymbol{X}_t\nabla f(\boldsymbol{x}_t) + \boldsymbol{X}_t\boldsymbol{A}^\top\boldsymbol{v}_t\|_\infty < 2\varepsilon$ and $\nabla f(\boldsymbol{x}_t) + \boldsymbol{A}^\top\boldsymbol{v}_t > 0$ for some $\boldsymbol{v}_t \in \mathbb{R}^m$.

During the iterations of Algorithm 2, the iterates $\boldsymbol{x}_t$ can be categorized into three types: (i) points that are not 2ε-KKT points; (ii) points that satisfy the 2ε-KKT conditions but not the $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 conditions; (iii) points that satisfy the $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 conditions.

By the initialization of $\boldsymbol{x}_0$, every $\boldsymbol{x} \in \Omega^\circ$ satisfies

$$\begin{aligned}\phi(\boldsymbol{x}_0) - \phi(\boldsymbol{x}) &= f(\boldsymbol{x}_0) - f(\boldsymbol{x}) + \varepsilon\sum_{i=1}^{n}\ln(x_i) - \varepsilon\sum_{i=1}^{n}\ln(x_{0,i}) \\ &\le f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + C_0\varepsilon.\end{aligned} \tag{71}$$

Since both accepted first-order IPTR steps and accepted negative-curvature steps decrease the potential function, this bounds the total potential decrease over all accepted steps.

For points of the first type, Lemma 6 guarantees that the first-order IPTR step decreases the potential function by at least $\varepsilon^2/(4l+4\varepsilon)$. Any accepted first-order IPTR step at a point of the second or third type also satisfies the same decrease condition. Consequently, the total number of accepted first-order IPTR steps is bounded by

$$T_1 = \frac{(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + C_0\varepsilon)(4l+4\varepsilon)}{\varepsilon^2}. \tag{72}$$

For points of the second type, if the first-order IPTR step satisfies the sufficient decrease condition, it is already included in the preceding count. Otherwise, invoking Proposition 5 with $\delta_0 = \delta/T$ finds a negative curvature direction $\widehat{\boldsymbol{e}}$ with probability at least $1-\delta_0$. According to Proposition 6, the corresponding step decreases the potential function by at least $\sqrt{\varepsilon^3}/(1024\rho^2)$. Therefore, the total number of accepted negative-curvature steps is bounded by

$$T_2 = \frac{1024\,(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + C_0\varepsilon)\,\rho^2}{\sqrt{\varepsilon^3}}. \tag{73}$$

Combining both cases, define $T = \max\{2T_2, 4T_1\}$, at most $T_1 < T/4$ iterations accept first-order IPTR steps, and at most $T_2 < T/2$ iterations accept negative-curvature steps. Since every nonterminal iteration accepts one of these two types of steps, the algorithm must terminate within T iterations. In this case, at least $T/4$ iterates belong to the third type, i.e., they are $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 points. Hence, Algorithm 2 finds a $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 point within T iterations with probability at least $1-\delta$.

It remains to bound the overall time complexity of Algorithm 2. We separately account for the cost of the first-order IPTR iterations and the negative-curvature finding procedure.

At each iteration t, we need to solve a linear programming problem subject to the equality constraint $\boldsymbol{A}\boldsymbol{X}_t\boldsymbol{d} = \boldsymbol{0}$ and the ball constraint $\|\boldsymbol{d}\| \leq \frac{\varepsilon}{l+2\varepsilon}$. This linear programming problem admits a closed-form solution as defined in (6).

Computing this solution requires the projection matrix $\boldsymbol{P}_t$ onto the null space of $\boldsymbol{A}\boldsymbol{X}_t$. To construct $\boldsymbol{P}_t$, we first compute the orthogonal basis $\boldsymbol{Z}_t$ of $\ker(\boldsymbol{A}\boldsymbol{X}_t)$. The time complexity of this step is $\mathcal{O}(nm^{\omega-1})$ due to the cost of rectangular matrix multiplication (Demmel et al., 2007), where ω is the matrix-multiplication exponent. Once $\boldsymbol{Z}_t$ is obtained, the projection matrix can be computed as $\boldsymbol{P}_t = \boldsymbol{Z}_t\boldsymbol{Z}_t^\top$. Subsequently, the computation of $\boldsymbol{d}_t$ involves a matrix-vector multiplication, which incurs a cost of $\mathcal{O}(mn)$. Updating $\boldsymbol{x}_{t+1}$ requires multiplying a diagonal matrix with a vector, with a time complexity of $\mathcal{O}(n)$. Therefore, each iteration of solving the interior point trust region problem has an overall time complexity of $\mathcal{O}(nm^{\omega-1})$. Since solving the first-order IPTR subproblem is invoked at most $\frac{(f(\boldsymbol{x}_0)-f(\boldsymbol{x}^*)+(C_0-1)\varepsilon)(4l+4\varepsilon)}{\varepsilon^2}$ times, the overall time complexity is bounded by $\mathcal{O}(nm^{\omega-1}/\varepsilon^2)$.

Then we bound the time complexity for negative curvature finding. In Procedure 1, the first step is to compute the projection matrix $\boldsymbol{P}$ onto $\ker(\boldsymbol{A}\boldsymbol{X})$, which requires $\mathcal{O}(nm^{\omega-1})$ time. In each subsequent iteration of Procedure 1, updating $\boldsymbol{y}^t$ only involves a matrix-vector multiplication, which takes $\mathcal{O}(mn)$ time. The algorithm proceeds for $\mathcal{T} = \frac{8l}{\sqrt{\varepsilon}}\log\left(\frac{8l}{\delta_0}\sqrt{\frac{n}{\pi\varepsilon}}\right)$ steps in total. Hence, a single invocation of Procedure 1 incurs a time complexity of $\widetilde{\mathcal{O}}(nm^{\omega-1} + mn/\sqrt{\varepsilon})$. According to Theorem 2, Procedure 1 will be invoked $\frac{1024(f(\boldsymbol{x}_0)-f(\boldsymbol{x}^*))\rho^2}{\sqrt{\varepsilon^3}}$ times with high probability. Therefore, the overall time complexity for calling Procedure 1 is $\widetilde{\mathcal{O}}(nm^{\omega-1}/\sqrt{\varepsilon^3} + mn/\varepsilon^2)$.

Combining both parts, the total time complexity is bounded by $\widetilde{\mathcal{O}}\left(nm^{\omega-1}/\varepsilon^2\right)$, which completes the proof.

C.4 Proof of Theorem 3

The proof follows the structure of Theorem 2. During the iterations, each iterate falls into one of the following three categories: (i) points that are not 2ε-KKT points; (ii) points that satisfy the 2ε-KKT conditions but not the $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 conditions; (iii) points that satisfy the $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 conditions.

By the initialization of $\boldsymbol{x}_0$, every $\boldsymbol{x} \in \Omega^\circ$ satisfies

$$\begin{aligned}\phi(\boldsymbol{x}_0) - \phi(\boldsymbol{x}) &= f(\boldsymbol{x}_0) - f(\boldsymbol{x}) + \varepsilon\sum_{i=1}^n \ln(x_i) - \varepsilon\sum_{i=1}^n \ln(x_{0,i}) \\ &\leq f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + C_0\varepsilon.\end{aligned} \tag{74}$$

Since both accepted first-order IPTR steps and accepted negative-curvature steps decrease the potential function, this bounds the total potential decrease over all accepted steps.

For points of the first type, Proposition 4 guarantees that the first-order IPTR step decreases the potential function by at least $\varepsilon^2/(2l + 4\varepsilon + 4)$. Any accepted first-order IPTR step at a point of the second or third type also satisfies the same decrease condition. Consequently, the total number of accepted first-order IPTR steps is bounded by

$$T_1 = \frac{(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + C_0\varepsilon)\,(2l + 4\varepsilon + 4)}{\varepsilon^2}. \tag{75}$$

For points of the second type, if the first-order IPTR step satisfies the sufficient decrease condition, it is already included in the preceding count. Otherwise, invoking Proposition 5 with $\delta_0 = \delta/T$ finds a negative curvature direction $\widehat{\boldsymbol{e}}$ with probability at least $1 - \delta_0$. According to Proposition 6, the corresponding step decreases the potential function by at least $\sqrt{\varepsilon^3}/(1024\rho^2)$. Since every accepted negative-curvature step satisfies this decrease condition, the total number of accepted

negative-curvature steps is bounded by

$$T_2 = \frac{1024\,(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + C_0\varepsilon)\,\rho^2}{\sqrt{\varepsilon^3}}. \tag{76}$$

Combining both cases, define

$$\begin{aligned} T &= \max\{2T_2, 4T_1\} \\ &= \max\left\{\frac{2048\,(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + C_0\varepsilon)\,\rho^2}{\sqrt{\varepsilon^3}}, \frac{8\,(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + C_0\varepsilon)\,(l + 2\varepsilon + 2)}{\varepsilon^2}\right\}. \end{aligned} \tag{77}$$

Under this bound, at most $T_1 \le T/4$ iterations accept first-order IPTR steps, and at most $T_2 \le T/2$ iterations accept negative-curvature steps. Therefore, if the algorithm runs for T iterations, at least $T/4$ iterates belong to the third type, Algorithm 3 finds a $(2\varepsilon, \sqrt{\varepsilon})$-KKT2 point within T iterations with probability at least $1 - \delta$.

In terms of gradient queries, the first-order IPTR subproblems require $\mathcal{O}(T+1)$ queries, while each negative curvature search requires $\widetilde{\mathcal{O}}(l/\sqrt{\varepsilon})$ queries. Since every invocation of Procedure 1 except the final one is followed by an accepted negative-curvature step, the number of invocations is at most $T_2 + 1$. Consequently, the overall number of gradient queries is bounded by

$$\mathcal{O}(T+1) + (T_2+1)\widetilde{\mathcal{O}}\left(\frac{l}{\sqrt{\varepsilon}}\right) = \widetilde{\mathcal{O}}\left(\frac{l\rho^2\,(f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*) + C_0\varepsilon + 1)}{\varepsilon^2}\right). \tag{78}$$

For the time complexity analysis, each call to Procedure 1 requires computing the projection matrix $\boldsymbol{P}_t$ onto the null space of $\boldsymbol{A}\boldsymbol{X}_t$, which incurs a cost of $\mathcal{O}(nm^{\omega-1})$. As shown earlier, the algorithm invokes Procedure 1 at most $\mathcal{O}(1/\varepsilon^{1.5})$ times. Therefore, the total time complexity contributed by all invocations of Procedure 1 is $\mathcal{O}(nm^{\omega-1}/\varepsilon^{1.5})$.

After each invocation of Procedure 1, once $\boldsymbol{P}_t$ has been computed, we restart the `SelectVector` procedure and recompute $\boldsymbol{R}_t$ based on the updated projection. By Proposition 3, the additional computational cost for T iterations of sparse updates is $\mathcal{O}(mnT)$. Since the total number of iterations between all Procedure 1 calls sums to $\mathcal{O}(1/\varepsilon^2)$, the total cost of sparsely updating $\boldsymbol{R}_t$ and computing $\boldsymbol{d}_t$ is $\mathcal{O}(mn/\varepsilon^2)$. Therefore, the overall time complexity is $\mathcal{O}(nm^{\omega-1}/\varepsilon^{1.5} + mn/\varepsilon^2)$.

D Auxiliary Lemmas

Lemma 7. *If $\boldsymbol{x} > \boldsymbol{0}$ and $\|\boldsymbol{d}\| \le \beta < 1$, then*

$$-\sum_{i=1}^{n} \ln(x_i + x_i d_i) + \sum_{i=1}^{n} \ln(x_i) \le -\boldsymbol{e}^\top \boldsymbol{d} + \frac{\beta^2}{2(1-\beta)}. \tag{79}$$

Proof. Since $\|\boldsymbol{d}\| \le \beta < 1$, we have $|d_i| \le \beta < 1$ for every i. For any $z \in [-\beta, \beta]$,

$$-\ln(1+z) + z = \int_0^z \frac{t}{1+t}\,dt \le \frac{z^2}{2(1-\beta)}. \tag{80}$$

Therefore,

$$\begin{aligned} -\sum_{i=1}^{n} \ln(x_i + x_i d_i) + \sum_{i=1}^{n} \ln(x_i) &= -\sum_{i=1}^{n} \ln(1+d_i) \\ &\le -\sum_{i=1}^{n} d_i + \frac{1}{2(1-\beta)} \sum_{i=1}^{n} d_i^2 \\ &\le -\boldsymbol{e}^\top \boldsymbol{d} + \frac{\beta^2}{2(1-\beta)}, \end{aligned} \tag{81}$$

where the last inequality follows from $\|\boldsymbol{d}\|^2 \le \beta^2$. □

Lemma 8 (Quadratic Upper Bound). *Under Assumption 1, for any* $\boldsymbol{x} \in \Omega^\circ$ *and* $\boldsymbol{d} \in \{\boldsymbol{d} : \|\boldsymbol{d}\| \leq \gamma, \boldsymbol{A}\boldsymbol{X}\boldsymbol{d} = 0\}$,

$$\|\boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\| \leq l, \tag{82}$$

and

$$f(\boldsymbol{X}(\boldsymbol{e}+\boldsymbol{d})) \leq f(\boldsymbol{x}) + \langle \boldsymbol{X}\nabla f(\boldsymbol{x}), \boldsymbol{d}\rangle + \frac{l}{2}\|\boldsymbol{d}\|^2. \tag{83}$$

Proof. Define $g(\boldsymbol{d}) := f(\boldsymbol{X}(\boldsymbol{e}+\boldsymbol{d}))$. Then $\nabla g(\boldsymbol{d}) = \boldsymbol{X}\nabla f(\boldsymbol{X}(\boldsymbol{e}+\boldsymbol{d}))$, and Assumption 1 ensures that ∇g is l-Lipschitz on the feasible displacement set. Consequently, $\|\nabla^2 g(\boldsymbol{d})\| \leq l$, and the standard quadratic upper bound gives

$$g(\boldsymbol{d}) \leq g(0) + \langle \nabla g(0), \boldsymbol{d}\rangle + \frac{l}{2}\|\boldsymbol{d}\|^2. \tag{84}$$

Rewriting these bounds in the original variables gives the result. □

Lemma 9 (Cubic Upper Bound). *Under Assumption 2, for any* $\boldsymbol{x} \in \Omega^\circ$ *and* $\boldsymbol{d} \in \{\boldsymbol{d} : \|\boldsymbol{d}\| \leq \gamma, \boldsymbol{A}\boldsymbol{X}\boldsymbol{d} = 0\}$,

$$f(\boldsymbol{X}(\boldsymbol{e}+\boldsymbol{d})) \leq f(\boldsymbol{x}) + \langle \boldsymbol{X}\nabla f(\boldsymbol{x}), \boldsymbol{d}\rangle + \frac{1}{2}\boldsymbol{d}^\top \boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}\boldsymbol{d} + \frac{\rho}{6}\|\boldsymbol{d}\|^3. \tag{85}$$

Proof. Let $g(\boldsymbol{d}) := f(\boldsymbol{X}(\boldsymbol{e}+\boldsymbol{d}))$. Assumption 2 implies that $\nabla^2 g$ is ρ-Lipschitz on the feasible displacement set. Therefore, the standard third-order Taylor bound gives

$$g(\boldsymbol{d}) \leq g(0) + \langle \nabla g(0), \boldsymbol{d}\rangle + \frac{1}{2}\boldsymbol{d}^\top \nabla^2 g(0)\boldsymbol{d} + \frac{\rho}{6}\|\boldsymbol{d}\|^3. \tag{86}$$

Substituting $\nabla g(0) = \boldsymbol{X}\nabla f(\boldsymbol{x})$ and $\nabla^2 g(0) = \boldsymbol{X}\nabla^2 f(\boldsymbol{x})\boldsymbol{X}$ gives the result. □