Noname manuscript No.
(will be inserted by the editor)

Complexity of a Projected Newton-CG Method for Optimization with Bounds

Yue Xie · Stephen J. Wright

Received: date / Accepted: date

Abstract This paper describes a method for solving smooth nonconvex minimization problems subject to bound constraints with good worst-case complexity and practical performance. The method contains elements of two existing methods: the classical gradient projection approach for bound-constrained optimization and a recently proposed Newton-conjugate gradient algorithm for unconstrained nonconvex optimization. Using a new definition of approximate second-order optimality parametrized by some tolerance ϵ (which is compared with related definitions from previous works), we derive complexity bounds in terms of ϵ for both the number of iterations required and the total amount of computation. The latter is measured by the number of gradient evaluations or Hessian-vector products. We also describe illustrative computational results on several test problems from low-rank matrix optimization.

Keywords Nonconvex Bound-constrained Optimization · Complexity Guarantees · Projected Gradient Method · Newton's Method · Conjugate Gradient Method

A preliminary version of this work has been archived in the workshop "Beyond First-Order Methods in ML Systems" at the 37th International Conference on Machine Learning, Vienna, Austria, 2020. Research is supported from NSF Awards 1740707, 1839338, 1934612, and 2023239; Subcontract 8F-30039 from Argonne National Laboratory; and Award N660011824020 from the DARPA Lagrange Program. This work was submitted when the first author was a postdoctoral research associate at the Wisconsin Institute for Discovery at University of Wisconsin-Madison.

Yue Xie
Department of Mathematics and Musketeers Foundation Institute of Data Science, The University of Hong Kong, Pokfulam, Hong Kong.
E-mail: yxie21@hku.hk

Stephen J. Wright
Computer Sciences Department, University of Wisconsin-Madison, 1210 W. Dayton St., Madison, WI, 53706.
E-mail: swright@cs.wisc.edu

Mathematics Subject Classification (2010) 49M15 · 68Q25 · 90C06 · 90C30 · 90C60

1 Introduction

We consider the problem

$$\min f(x) \quad \text{subject to } x \in \Omega, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable and is bounded below by $f_{\text{low}} > -\infty$ on the closed feasible set Ω. We focus on Ω defined by nonnegativity constraints on a subset $\mathcal{I}$ of the variables, that is,

$$\Omega \triangleq \{x \in \mathbb{R}^n \mid x^i \geq 0,\ i \in \mathcal{I}\}, \quad \text{where } \mathcal{I} \subseteq \{1, 2, \ldots, n\}. \tag{2}$$

Bounds are the simplest type of inequality constraint. Euclidean projection onto the feasible set Ω, a trivial operation when Ω is defined by bounds, is a fundamental component of several successful algorithms. Bound-constrained subproblems often arise in algorithms for more complicated constrained optimization problems, such as augmented Lagrangian methods. Bound constraints also appear in machine learning problems such as nonnegative least-squares and nonnegative matrix factorization [13]. Approaches of several types have been proposed for solving this problem, including gradient projection, active set methods, and interior-point methods.

In this paper, we describe a line-search method for solving (1), (2) that exploits the simplicity of Euclidean projection onto Ω. It combines gradient projection with a Newton-conjugate gradient (Newton-CG) method for smooth nonconvex unconstrained optimization proposed recently in [23]. The elements of our method are well known for their good practical performance in various optimization contexts. By combining these elements in the right way, and introducing judicious strategies for diagonal scaling, step length acceptance, and detection of negative curvature, we equip the method with a worst-case complexity theory that matches best-known theoretical bounds for bound-constrained optimization and even for unconstrained optimization. Preliminary numerical results confirm that the method has appealing practical performance. In contrast to most previous works on complexity, we prove results for both iteration and *computational* complexity. The latter is measured in terms of two key operations: evaluation of a gradient at a given point, and computation of a Hessian-vector product involving an arbitrary vector. (The latter is known to cost a modest multiple of a gradient evaluation when computational differentiation techniques are used [16].) Our methods does not require explicit calculation or storage of the Hessian; it accesses the Hessian only via products with given vectors.

Background and Prior Work. There has been renewed interest in devising optimization algorithms with worst-case complexity guarantees for constrained nonconvex optimization. Interior-point type methods were developed to solve nonconvex problems with bound constraints [4], or with bounds and linear equality constraints [17]. A log-barrier method for bound-constrained problems was proposed in [22]. Like the present paper, this method made use of the Newton-CG method of [23], but in a quite different way. An adaptive cubic regularization algorithm was proposed in [7] to solve nonconvex optimization with general convex constraints. Later, in [8], the authors of [7] designed a novel two-phase target-following algorithm to address a more general problem class: nonconvex optimization with nonlinear equality constraints and a general convex feasible region. They also generalize the concept of approximate first-order optimal point to arbitrary high-order and apply a conceptual high-order algorithm for obtaining such a point [9]. Authors of [6] outline a high-order algorithm that obtains approximate first-order optimal point of a nonconvex optimization problem with general constraints. A Hessian barrier algorithm was recently proposed [11], based on self-concordant barrier functions, which solves nonconvex problems with general conic constraints and linear equality constraints.

In most of these articles, good complexity results follow from the use of the Hessian and sometimes higher-order derivatives: $\mathcal{O}(\epsilon^{-3/2})$ iteration/evaluation[1] complexity to locate an ϵ-approximate first-order optimal point [7, 8,6] or a $(\epsilon, \sqrt{\epsilon})$ second-order optimal point [4,17,22,11]. (Here ϵ and $\sqrt{\epsilon}$ represent the precision of first- and second-order optimality conditions, respectively.) The qth-order algorithm in [9] locates an ϵ-approximate qth-order solution in $\mathcal{O}(\epsilon^{-(q+1)})$ iterations, while the qth-order algorithm in [6] finds an ϵ-approximate first-order solution in $\mathcal{O}(\epsilon^{-(q+1)/q})$ iterations.

Complexity results in the works discussed above focus on iteration/evaluation complexity; less attention is paid to the bounds on the total amount of computation required. In fact, these methods can require solution of nonconvex subproblems that may themselves require a significant and undetermined amount of computation. For example, in [7,8], a potentially expensive and complicated cubic regularized subproblem (itself a constrained nonconvex problem) needs to be solved to approximate first-order optimality at each iteration, while the higher-order methods of [9] and [6] require solution of complicated subproblems involving higher-order derivatives. Moreover, implementations of these methods may require explicit evaluation of the Hessian or higher-order derivatives. The method of this paper, by contrast, requires explicit evaluation only of gradients; the Hessians are accessed only via Hessian-vector products. This fact allows us to define meaningful bounds on computational complexity.

The pursuit of optimal iteration/evaluation complexity results may compromise the practicality of algorithms. For example, subproblems in the second-order algorithms from [4] and [17] have a small trust-region radius that depends

[1] Iteration complexity in this paper is a bound on the number of outer iterations in an algorithm. It is equivalent to evaluation complexity (a count of the number of evaluations of gradients, Hessians, or higher-order derivatives) for purposes of this discussion.

on ϵ. The log-barrier approach of [22] has unimpressive practical performance, as we see in Section 5.

Other works that address complexity of constrained nonconvex optimization, include [10], which discusses the trust funnel algorithm to solve optimization with equality constraints; [26,15,5,25], which discuss augmented Lagrangian methods (ALM); and [20], concerning penalty methods. In [15], ALM and appropriate first-order algorithms to solve subproblems are utilized to locate ϵ approximate first-order point, with evaluation complexity arbitrarily close to $\mathcal{O}(\epsilon^{-4})$. Complexity of a safeguarded ALM is derived in [5] to find first-order stationary points, but the cost of solving the subproblems is not well defined. In [20], complexity results are established in terms of the number of proximal gradient steps needed to find an ϵ first-order stationary points. The complexity can be improved to $\mathcal{O}(\epsilon^{-5/2})$ (omitting logarithm terms) when the constraint functions are convex and Slater's condition holds. [10,26,25] consider optimization with equality constraints that do not accommodate the bound-constrained problem class (1), (2).

A complicating factor in comparing complexity of methods for finding approximate optimal points is that the definitions of such points vary between papers. This is not unexpected since different papers consider a variety of constraint types, and the approximate optimality conditions are adapted to the particular formulations. The relation between different definitions has not been discussed in any detail, even for the case of optimization with bounds. We believe that a proper discussion facilitates a better understanding of the goals and characteristics of different algorithms.

Approach and Contributions. We describe an algorithm for locating an approximate second-order point of the problem (1),(2) that has good worst-case complexity bounds — similar to the unconstrained case ($\Omega = \mathbb{R}^n$ in (1)) — and is also practical.

As a preliminary to our description of the algorithm, we state our definition of approximate second-order optimality, alongside four other definitions that have appeared in the literature. These definitions are typically parametrized by a tolerance ϵ. We introduce a second parameter p that represents the power of ϵ that determines the approximate condition involving the Hessian, and refer to the resulting conditions as "(ϵ, p)-second-order optimality" or "(ϵ, p)-2o" for short. The alternative definitions that we discuss in this article are based on those from [9,17,22,4], specialized to the bound-constrained problem (1),(2), with $\mathcal{I} = \{1, \ldots, n\}$. We make comparisons among all these definitions, using a new notion of "essentially stronger".

Practical methods that make use of gradient projection and Newton scaling have yet to be considered seriously as methods with good complexity guarantees for bound-constrained problems. Such methods exploit the simplicity of the projection operation for Ω in (2), as well as the benefits of second-order information that have been shown in the unconstrained context. The two-metric projection framework proposed by Bertsekas [1,2] provides a potential framework, for appropriate choice of scaling matrix. This method takes steps of the

form

$$x_{k+1} \triangleq P(x_k - \alpha_k D_k \nabla f(x_k)), \tag{3}$$

where D_k is a symmetric positive definite matrix (with a certain structure defined below) and $P(z)$ is the projection onto the feasible set Ω in (2), defined by

$$[P(z)]^i = \begin{cases} \max\{z^i, 0\} & i \in \mathcal{I}, \\ z^i & \text{otherwise.} \end{cases} \tag{4}$$

The matrix D_k scales the free and active parts of the gradient differently, in a way that guarantees decrease in the objective function for sufficiently small positive steplengths α_k. Denoting a set of "apparently-active" components of x_k by

$$I_k^+(\epsilon_k) \triangleq \{i \in \mathcal{I} \mid 0 \le x_k^i \le \epsilon_k,\ \nabla_i f(x_k) > 0\}, \tag{5}$$

for small positive ϵ_k, D_k is assumed to be positive diagonal in the $I_k^+(\epsilon_k)$ components, that is, $D_k[i,j] = 0$ if either i or j is in $I_k^+(\epsilon_k)$ with $j \neq i$, and $D_k[i,i] > 0$ for all $i \in I_k^+(\epsilon_k)$. The two-metric projection method can have rapid convergence when $f(x)$ is convex and the square submatrix of D_k for the "apparently-free" indices $i \notin I_k^+(\epsilon_k)$ is derived from the corresponding submatrix of the Hessian $\nabla^2 f(x_k)$. The complexity properties of this method in the setting of nonconvex f are the subject of ongoing work.

Inspired by both two-metric gradient projection approach and the Newton-CG algorithm for unconstrained optimization described in [23], we propose a projected Newton-CG algorithm. We show that the algorithm terminates within $\mathcal{O}(\epsilon^{-3/2})$ iterations and outputs an $(\epsilon, \frac{1}{2})$-2o point with high probability. In each iteration of the projected Newton-CG, we either (1) take a gradient projection step; (2) take a projected Newton-CG step, obtained via a capped CG procedure applied to the apparently-free components, or (3) take a projected step along a negative curvature direction of a diagonally scaled Hessian. The operations required to calculate each type of step are well defined, and are similar to those used in [23,24]. These "fundamental operations" are of two types: (1) a gradient calculation, and (2) computation of the product of the Hessian with an arbitrary vector — an operation that does not require explicit computation or knowledge of the Hessian and that can be performed at roughly equivalent cost to a gradient evaluation; see [16]. Apart from these fundamental operations, the only significant contributors to computational cost are operations involving vectors of length n (inner products and saxpys), whose $O(n)$ cost is dominated by the cost of the fundamental operations for all functions of interest. By contrast, other methods require solution of potentially expensive constrained nonconvex subproblems in each iteration [6–9] and possibly explicit evaluation of Hessians and higher derivatives, making it difficult to provides meaningful bounds on computational complexity.

Table 1 shows iteration/evaluation complexity and operation complexity results for our algorithm (last row) and existing algorithms, based on their respective definitions of (ϵ, p)-2o. The "operation complexity" results are upper

Table 1 Complexity estimates for nonconvex optimization procedures involving bounds.

Definition of (ϵ, p)-2o*	Iteration/evaluation Complexity	Operation Complexity ($p = \frac{1}{2}$)	Ref.
(12)	$\mathcal{O}(\epsilon^{-3})^{\star\star}$ $(p = 1)$	–	[9]
(13)	$\mathcal{O}(\epsilon^{-3/2})$ $(p = 1/2)$	–	[17]
(14)	$\tilde{\mathcal{O}}(n\epsilon^{-1/2} + \epsilon^{-3/2})^{\dagger}$ $(p = 1/2)$	$\tilde{\mathcal{O}}(n\epsilon^{-3/4} + \epsilon^{-7/4})$, n large $\tilde{\mathcal{O}}(n\epsilon^{-3/2})$, n small	[22]
(15)	$\mathcal{O}(\epsilon^{-3/2})$ $(p = 1/2)$	–	[4]
(9) w. $\mathcal{I} = \{1, \ldots, n\}$	$\mathcal{O}(\epsilon^{-3/2})$ $(p = 1/2)$	$\mathcal{O}(\epsilon^{-3/2} \min\{n, \epsilon^{-1/4} \log(\frac{n}{\epsilon\delta})\})$	(here)

$*$: Definition of (ϵ, p)-2o is based on the paper in "Ref." but tailored to problem (1),(2) with $\mathcal{I} = \{1, \ldots, n\}$.
$\star\star$: When $p = 1$, accuracy on the optimality condition involving Hessian is higher, leading to a higher complexity bound.
†: $\tilde{\mathcal{O}}$ represents $\mathcal{O}$ with logarithmic factors omitted.

bounds on the number of fundamental operations required to find an approximate solution.

Illustrative numerical experiments on nonnegative matrix factorization problems show that the projected Newton-CG algorithm has good practical performance: It contends well with gradient projection method and the log-barrier Newton-CG algorithm proposed in [22], and is competitive with approaches that are specialized to this problem in relatively low dimensions.

With minor modifications (c.f. Appendix D), the projected Newton-CG can be applied to problems with two-sided bounds, where Ω is redefined as $\{x \in \mathbb{R}^n \mid 0 \leq x^i \leq u^i, i \in \mathcal{I}\}$, $\mathcal{I} \subseteq \{1, 2, \ldots, n\}$, with the same complexity guarantees.

Organization. In Section 2, we introduce some basic assumptions and definitions to be used throughout the article. Definitions of the approximate second-order optimal point in our work and others are discussed in Section 3. The projected Newton-CG is presented and analyzed in Section 4. Section 5 describes numerical experiments. Section 6 contains some concluding remarks.

We include in the Appendix details of the relationship between different definitions of approximate second-order optimality, the oracles utilized in the projected Newton-CG algorithm, and extension to two-sided bounds.

2 Preliminaries

We summarize here some notations, two assumptions used throughout the paper, and (exact) optimality conditions for (1), (2).

Notation. We use subscripts for iteration numbers (usually k) throughout, and denote components of vectors by superscripts and components of matrices

using square-bracket notation, with $[i, j]$ denotes the i, j element. We use the following notation for gradient and Hessian of f at x_k:

$$g_k \triangleq \nabla f(x_k), \quad H_k \triangleq \nabla^2 f(x_k).$$

We use $\nabla_i f(x)$ to denote the ith component of $\nabla f(x)$. diag(v) is a diagonal matrix with v^i being its $[i, i]$ element. sgn$(z) = 1$ if $z \geq 0$ and sgn$(z) = -1$ otherwise. $\|\cdot\|$ denotes the 2-norm of a vector or a matrix. $c_+ \triangleq \max\{c, 0\}$ for a scalar $c \in \mathbb{R}$. $\mathcal{I}^c \triangleq \{1, \ldots, n\} \setminus \mathcal{I}$. $P(\cdot)$ denotes the projection onto the feasible region Ω.

Assumptions. The following assumptions are used throughout the paper, though they are not mentioned explicitly in the statements of some lemmas.

Assumption 1 *The level set $\mathcal{L}_f(x_0) \triangleq \{x \in \mathbb{R}^n \mid x \in \Omega,\ f(x) \leq f(x_0)\}$ is compact.*

Assumption 2 *f is twice Lipschitz continuously differentiable on an open convex set containing $\mathcal{L}_f(x_0)$ and all the trial points generated by the algorithms proposed below.*

Lipschitz constants for f, $\nabla f(x)$ and $\nabla^2 f(x)$ on the set described in Assumption 2 are denoted by L_f, L_g and L_H, respectively. Thus, for any $x, v \in \mathbb{R}^n$ such that x and $x + v$ are in this set, we have

$$f(x + v) \leq f(x) + L_f \|v\|, \tag{6a}$$

$$f(x + v) \leq f(x) + \nabla f(x)^T v + \frac{L_g}{2}\|v\|^2, \tag{6b}$$

$$f(x + v) \leq f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v + \frac{L_H}{6}\|v\|^3. \tag{6c}$$

Therefore, $\|\nabla f(x)\| \leq L_f$ and $\|\nabla^2 f(x)\| \leq L_g$ over $\mathcal{L}_f(x_0)$.

Optimality Conditions. We can write first-order optimality conditions for (1), (2) (also known as stationarity conditions) at a point $\bar{x}$ as follows:

$$\begin{aligned} &\bar{x}^i \geq 0, \quad \nabla_i f(\bar{x}) \geq 0, \quad \forall i \in \mathcal{I}; \\ &\nabla_i f(\bar{x}) = 0, \quad \forall i \in \mathcal{I}^c \cup \{i \in \mathcal{I} \mid \bar{x}^i > 0\}. \end{aligned} \tag{7}$$

A weak second-order condition for (1), (2) is that the two-sided projection of $\nabla^2 f(\bar{x})$ onto the variables i such that $\bar{x}^i > 0$ or $i \in \mathcal{I}^c$ is positive semidefinite, which is equivalent to

$$z^T \nabla^2 f(\bar{x}) z \geq 0, \quad \forall z \in \{z \in \mathbb{R}^n \mid z^i = 0, i \in \{i \in \mathcal{I} \mid \bar{x}^i = 0\}\}. \tag{8}$$

This condition coincides with the usual second-order necessary condition where there are no "degenerate" indices, that is, indices $i \in \mathcal{I}$ for which both $\bar{x}^i = 0$

and $\nabla_i f(\bar{x}) = 0$. When such indices exist, a standard second-order necessary condition is:

$$z^T \nabla^2 f(\bar{x}) z \geq 0, \quad \forall z \in \left\{ z \in \mathbb{R}^n \,\middle|\, \begin{array}{l} z^i = 0, \text{ if } i \in \mathcal{I}, \bar{x}^i = 0, \nabla_i f(\bar{x}) > 0, \\ z^i \geq 0 \text{ if } i \in \mathcal{I}, \bar{x}^i = 0, \nabla_i f(\bar{x}) = 0. \end{array} \right\}.$$

However, checking this condition can be as hard as checking copositivity of a matrix, which is NP-hard. Thus, as in previous works (such as [22]), we base our analysis on the less stringent condition (8).

3 Approximate second-order optimal points

In this section we give our definition of (ϵ, p)-approximate second-order optimal points and compare it with similar definitions in the literature.

Our definition of an (ϵ, p)-2o point is as follows.

Definition 1 ((ϵ, p)-2o, Def1) For $\epsilon, p > 0$, x is an (ϵ, p)-2o point of (1),(2) according to **Def1** if $x \in \Omega$ and for sets J^+ and J^- defined by

$$\begin{aligned} J^+ &\triangleq \{i \in \mathcal{I} \mid 0 \leq x^i \leq \sqrt{\epsilon}\}, \\ J^- &\triangleq \{1, \ldots, n\} \setminus J^+ = \mathcal{I}^c \cup \{i \in \mathcal{I} \mid x^i > \sqrt{\epsilon}\}, \end{aligned}$$

and for diagonal matrix $S = \text{diag}(s)$ with $s^i = 1$ when $i \in J^-$ and $s^i = x^i$ when $i \in J^+$, we have

$$\|S \nabla f(x)\| \leq 2\epsilon, \quad \nabla_i f(x) \geq -\epsilon^{3/4}, \text{ for all } i \in J^+, \tag{9a}$$

$$S \nabla^2 f(x) S \succeq -\epsilon^p I. \tag{9b}$$

Definition 1 is motivated by the (weak) second-order optimal conditions (7) and (8). In fact, if we let $\epsilon = 0$, then $(0, p)$-2o satisfies (7) and (8) exactly. The following lemma further justifies Definition 1 and our purpose to find an (ϵ, p)-2o given small ϵ.

Lemma 1 *Consider problem* (1),(2). *Suppose we have a positive scalar sequence* $\{\epsilon_k\}$ *with* $\epsilon_k \downarrow 0$ *and vector sequence* $\{x_k\} \subseteq \Omega$ *with* $x_k \to x^*$ *such that* x_k *is* (ϵ_k, p)*-2o according to Definition 1. Then* x^* *satisfies second-order optimal conditions* (7), (8). *That is, for sets* $\mathcal{J}_*^-$ *and* $\mathcal{J}_*^-$ *defined by*

$$\mathcal{J}_*^- \triangleq \mathcal{I}_c \cup \{i \in \mathcal{I} \mid (x^*)^i > 0\}, \quad \mathcal{J}_*^+ \triangleq \{1, 2, \ldots, n\} \setminus \mathcal{J}_*^-,$$

we have

$$(x^*)^i \geq 0, \quad \nabla_i f(x^*) \geq 0, \quad \forall i \in \mathcal{I}; \tag{10a}$$

$$\nabla_i f(x^*) = 0, \quad \forall i \in \mathcal{J}_*^-; \tag{10b}$$

$$z^T \nabla^2 f(x^*) z \geq 0, \quad \forall z \in \{z \in \mathbb{R}^n \mid z^i = 0,\, i \in \mathcal{J}_*^+\}. \tag{10c}$$

Proof Denote sets $\mathcal{J}_k^+$, $\mathcal{J}_k^-$ and diagonal matrix $\mathcal{S}_k = \mathrm{diag}(s_k)$ which correspond to J^+, J^-, and S in Definition 1 with $x = x_k$, $\epsilon = \epsilon_k$ and $s = s_k$. Note that since $x_k \to x^*$ and $\epsilon_k \downarrow 0$, there exists $\bar{k}$ such that for any $k > \bar{k}$, we have $\mathcal{J}_k^+ \subseteq \mathcal{J}_*^+$, $\mathcal{J}_*^- \subseteq \mathcal{J}_k^-$. Our claim that x^* satisfies (10) is a consequence of the following four observations.

(i) Feasibility of x^* follows from closedness of Ω.

(ii) For any $i \in \mathcal{I}$ and any k, either $i \in \mathcal{J}_k^+$ so $\nabla_i f(x_k) \geq -\epsilon_k^{3/4}$, or $i \in \mathcal{J}_k^-$ so $|\nabla_i f(x_k)| \leq 2\epsilon_k \implies \nabla_i f(x_k) \geq -2\epsilon_k$. By taking limits, we have $\nabla_i f(x^*) \geq 0$.

(iii) Fix any $i \in \mathcal{J}_*^-$. For all $k > \bar{k}$, we have $i \in \mathcal{J}_k^-$. Therefore, $s_k^i = 1$ and $|\nabla_i f(x_k)| \leq 2\epsilon_k$. By taking limits, we have $\nabla_i f(x^*) = 0$.

(iv) Fix any $z \in \{z \in \mathbb{R}^n \mid z^i = 0, i \in \mathcal{J}_*^+\}$. For all $k > \bar{k}$, we have $i \in \mathcal{J}_k^+ \implies i \in \mathcal{J}_*^+ \implies z^i = 0$, so that $\mathcal{S}_k z = z$. Since $z^T \mathcal{S}_k \nabla^2 f(x_k) \mathcal{S}_k z \geq -\epsilon_k^p \|z\|^2$ for any k, we have by taking limits that $z^T \nabla^2 f(x^*) z \geq 0$.

□

We now identify several definitions of approximate second-order optimal conditions proposed in literature and discuss their relationship. For simplicity of notation, we use (ϵ, p)-2o to denote (ϵ, p)-approximate second-order optimal point. We assume $\epsilon, p > 0$ throughout. For simplicity, we assume in this subsection that

$$\mathcal{I} \triangleq \{1, 2, \ldots, n\}, \tag{11}$$

(so that $\Omega = \mathbb{R}^n_+$, the nonnegative orthant). When we refer to Definition 1 or **Def1** in this subsection, we implicitly assume that (11) holds.

We start from a definition in [9], which is defined for optimization with general convex constraints and high-order optimal points. Here we tailor it to fit the scope of this paper: second-order optimal points and bound-constrained optimization: (1), (2), (11).

Definition 2 ([9], Def2) x is an (ϵ, p)-2o of (1), (2), (11) according to **Def2** if $x \geq 0$ and, for some user-defined constant $\Delta_{\max}$ that is independent of x and ϵ, there exists $\Delta \in (0, \Delta_{\max}]$ such that

$$\begin{aligned} \left|\mathrm{globalmin}_{x+d\in\Omega, \|d\|\leq\Delta} \quad \nabla f(x)^T d\right| &\leq \Delta\epsilon, \\ \left|\mathrm{globalmin}_{x+d\in\Omega, \|d\|\leq\Delta} \quad \nabla f(x)^T d + \frac{1}{2} d^T \nabla^2 f(x) d\right| &\leq \Delta^2 \epsilon^p. \end{aligned} \tag{12}$$

$\Delta_{\max}$ is often chosen to reduce the effort in global minimization.

The following three definitions are from [17,22,4] tailored to our problem of interest. Here we let $X = \mathrm{diag}(x)$, $\bar{X} = \mathrm{diag}(\min\{x, \mathbf{1}\})$ and $\mathbf{1}$ denotes the vectors with all elements being 1.

Definition 3 ([17], Def3) x is an (ϵ, p)-2o of (1), (2), (11) according to **Def3** if

$$\begin{aligned} x \geq 0,\ \nabla f(x) \geq -\epsilon \mathbf{1},\ \|X \nabla f(x)\|_\infty \leq \epsilon, \\ X \nabla^2 f(x) X \succeq -\epsilon^p I_n. \end{aligned} \tag{13}$$

Definition 4 ([22], Def4) x is an (ϵ, p)-2o of (1), (2), (11) according to **Def4** if

$$\begin{aligned} x \geq 0,\ \nabla f(x) \geq -\epsilon \mathbf{1},\ \|\bar{X}\nabla f(x)\|_\infty \leq \epsilon, \\ \bar{X}\nabla^2 f(x)\bar{X} \succeq -\epsilon^p I_n. \end{aligned} \tag{14}$$

Definition 5 ([4], Def5) x is an (ϵ, p)-2o of (1), (2), (11) according to **Def5** if

$$\begin{aligned} x \geq 0,\ \|X\nabla f(x)\|_\infty \leq \epsilon, \\ X\nabla^2 f(x)X \succeq -\epsilon^p I_n. \end{aligned} \tag{15}$$

The relationships between each of these definitions and second-order criticality has been discussed in the respective works. In order to discuss the relation between any two of these definitions including ours, we propose the following concept, which relates pairs of definitions of (ϵ, p)-2o under the assumption that x is confined to a compact set $\mathcal{X}$.

Definition 6 We say that **DefA** is **essentially stronger** than **DefB** on $\mathcal{X}$ if given any sufficiently small $\epsilon \in (0, 1]$, any (ϵ, p)-2o $x \in \mathcal{X}$ by **DefA** is also a $(c\epsilon, p)$-2o by **DefB**, where $c > 0$ is a constant independent of ϵ or x. We denote this relation as $\mathbf{DefA} \gtrsim_{f,\mathcal{X},p} \mathbf{DefB}$, simplified as $\mathbf{DefA} \gtrsim \mathbf{DefB}$. We say that **DefA** and **DefB** are **essentially equivalent** (denoted $\mathbf{DefA} \approx \mathbf{DefB}$) if $\mathbf{DefA} \gtrsim \mathbf{DefB}$ and $\mathbf{DefB} \gtrsim \mathbf{DefA}$.

Transitivity of the relation $\gtrsim$ is shown in Lemma 6.

Comparison and evaluation of complexity of different algorithms makes more sense if we are able to relate the guarantees on the points they produce according to the relations in Definition 6. In fact, if we care most about the complexity as a function of the accuracy parameter ϵ, Definition 6 is natural and intuitive due to the following theorem.

Theorem 1 *Given any $\epsilon > 0$ sufficiently small, suppose that an algorithm can find an (ϵ, p)-2o $x \in \mathcal{X}$ by* **DefA** *in $\mathcal{O}(\epsilon^{-q})$ iterations ($q > 0$) and* $\mathbf{DefA} \gtrsim \mathbf{DefB}$. *Then the algorithm can also locate an (ϵ, p)-2o by* **DefB** *in $\mathcal{O}(\epsilon^{-q})$ iterations.*

Proof Since $\mathbf{DefA} \gtrsim \mathbf{DefB}$, there is a constant $c > 0$ such that for all $\epsilon > 0$ sufficiently small, an $(\epsilon/c, p)$-2o by **DefA** is an (ϵ, p)-2o by **DefB**. By assumption, the algorithm can locate $(\epsilon/c, p)$-2o by **DefA** in $\mathcal{O}((\epsilon/c)^{-q}) = \mathcal{O}(\epsilon^{-q})$ number of iterations. The result follows. □

We can now clarify several pairwise relations between the Definitions 1-5. The proof of the following result appears in Appendix A.

Theorem 2 *Suppose that $\mathcal{X}$ is a compact set. Then we have the following.*

(1) $\mathbf{Def2} \gtrsim \mathbf{Def3}$.
(2) $\mathbf{Def3} \approx \mathbf{Def4}$.
(3) $\mathbf{Def4} \gtrsim \mathbf{Def5}$.

(4) **Def1** $\gtrsim$ **Def5**.

The assumption in Theorem 2 on compactness of $\mathcal{X}$ is mild. In fact, many works in literature assume that the iterates generated by their algorithms lie in a compact region, for example, the sublevel set of the objective function. By Theorem 2, we have the following relation chart of Definition 1-Definition 5:

$$\textbf{Def2} \gtrsim \textbf{Def3} \approx \textbf{Def4} \gtrsim \textbf{Def5}, \quad \textbf{Def1} \gtrsim \textbf{Def5}.$$

Note that each $\gtrsim$ relation above is probably strict. For example, **Def2** considers the global minimum of the first-order and second-order Taylor expansions of f over a small trust region, while **Def3** (in fact all other definitions) is only closely related to the *weak* second-order necessary conditions (7),(8) for x being a local minimal point. **Def5** is weaker than others since it does not offer an appropriate lower bound on $\nabla_i f(x)$ when $x^i = 0$. In fact, the relation between **Def5** and second-order criticality is also weaker than others. Unfortunately, we cannot describe by $\gtrsim$ the relation between our definition (**Def1**) with definitions other than **Def5**. On one hand, the condition $\nabla_i f(x) \geq -\epsilon^{-3/4}, i \in J^+$ in **Def1** is weaker; on the other hand, the condition $\|S\nabla f(x)\| \leq 2c\epsilon$ in **Def1** is strong and cannot be implied by other (ϵ, p)-2o definitions for any constant c independent of ϵ. An illustrative example is given in Appendix A, Example 1.

4 Projected Newton-CG method and its complexity

We now describe a projected Newton-CG algorithm to find an $(\epsilon, \frac{1}{2})$-2o point according to Definition 1 for problem (1), (2), and analyze its complexity properties.

4.1 Description of the Algorithm

Given the sequence of iterates $\{x_k\}$ and a positive scalar sequence $\{\epsilon_k\}$ we define the following index sets inspired by the two-metric projection method (3), (5):

$$\begin{aligned} J_k^+ &\triangleq \{i \in \mathcal{I} \mid 0 \leq x_k^i \leq \epsilon_k\}, \\ J_k^- &\triangleq \{1, \ldots, n\} \setminus J_k^+ = \mathcal{I}^c \cup \{i \in \mathcal{I} \mid x_k^i > \epsilon_k\}. \end{aligned} \tag{16}$$

Let g_k^-, H_k^- be the subvector and square submatrix of g_k and H_k, resp., corresponding to index set J_k^-. Similarly, we use g_k^+ and H_k^+ for the subvector and square submatrix of g_k and H_k, resp., corresponding to index set J_k^+. For search direction d_k, denote d_k^- and d_k^+ in the same fashion. Define the scaling vector s_k and diagonal scaling matrix S_k as follows:

$$s_k^i \triangleq \begin{cases} x_k^i, & i \in J_k^+ \\ 1, & i \in J_k^- \end{cases}, \quad S_k \triangleq \operatorname{diag}(s_k). \tag{17}$$

We can then define the projected Newton-CG algorithm as Algorithm 1.

Algorithm 1 Projected Newton-CG (PNCG)

(Initialization) Choose an initial point $x_0 \geq 0$, tolerance $\epsilon_g > 0$, scalar sequence $\{\epsilon_k\}$ with $\epsilon_k \in (0,1)$ for all k, backtracking parameters $\theta \in (0,1)$, accuracy parameter $\zeta \in (0,1)$, step acceptance parameter $\eta \in \left(0, \frac{1-\zeta}{2}\right)$.

for $k = 0, 1, 2, \ldots$ **do**

  if $J_k^+ \neq \emptyset$ **and** ($g_k^i < -\epsilon_k^{3/2}$ for some $i \in J_k^+$ **or** $\|S_k^+ g_k^+\| > \epsilon_k^2$) **then**

    (Gradient Projection step) Let $d_k := -g_k$;

    Let $\tilde{m}_k$ be the smallest nonnegative integer such that

$$f(P(x_k + \theta^m d_k)) < f(x_k) - \frac{1}{2}(x_k - P(x_k + \theta^m d_k))^T g_k;$$

    Let $x_{k+1} := P(x_k + \theta^{\tilde{m}_k} d_k)$;

  else if $J_k^- \neq \emptyset$ **and** $\|g_k^-\| > \epsilon_g$ **then**

    (Newton-CG step) Call Algorithm 3 (Capped CG, Appendix B) with $H := H_k^-$, $\epsilon := \epsilon_k$, $g := g_k^-$, accuracy parameter ζ and upper bound M on Hessian norm (if provided). Obtain outputs $t \in \mathbb{R}^{|J_k^-|}$ and d_type;

    if $d\_\text{type} = \text{NC}$ **then**

      Let $d_k^- := -\text{sgn}(t^T g_k^-) \frac{|t^T H_k^- t|}{\|t\|^2} \frac{t}{\|t\|}$; (Negative curvature direction)

    else

      Let $d_k^- := t$; (Approx. solution to reduced Newton equations)

    end if

    Let $d_k^+ = 0$ (Complete d_k with zeros in the active components)

    Let m_k be smallest nonnegative integer such that

$$f(P(x_k + \theta^m d_k)) < f(x_k) - \eta\theta^{2m}\epsilon_k\|d_k\|^2;$$

    Let $\alpha_k := \theta^{m_k}$, $x_{k+1} := P(x_k + \theta^{m_k} d_k)$;

  else

    Call Procedure 4 (Minimum Eigenvalue Oracle (MEO), Appendix C) with $H := S_k H_k S_k$, $\epsilon := \epsilon_k$ and the upper bound of norm of H if known.

    if Procedure 4 certifies that $S_k H_k S_k \succeq -\epsilon_k I$ **then**

      STOP and output x_k;

    end if

    (Negative curvature step) Let $d_k := -\text{sgn}(g_k^T S_k d) \cdot |d^T S_k H_k S_k d| \cdot d$, where d is the output of Procedure 4;

    Let $\bar{m}_k$ be the smallest nonnegative integer such that

$$f(P(x_k + \theta^m S_k d_k)) < f(x_k) - \eta\theta^{2m}\|d_k\|^3.$$

    Let $x_{k+1} := P(x_k + \theta^{\bar{m}_k} S_k d_k)$;

  end if

end for

Elements of Algorithm 1. As in the two-metric projection method (3), our method starts each iteration by partitioning the components of x into the "apparently-free" and "apparently-active" indices based on their proximity to the boundary and a threshold parameter ϵ_k. Then one of three types of steps is taken. For all such steps, backtracking in combination with projection onto the feasible set is used to determine an appropriate steplength.

– **Gradient projection step:** If examination of the gradient components corresponding to the apparently-active components indicate that a signif-

icant improvement in f can be obtained by taking a standard gradient projection step, such a step is taken.
- **Newton-CG step on apparently-free components:** When the gradient corresponding to the apparently-free components is above the threshold ϵ_g, the Capped CG procedure (c.f. Appendix B) is called to either find an approximate Newton step in these components, or else return a direction of negative curvature. Only the apparently-free components are modified in a step of this type.
- **Scaled negative curvature step (full-dimensional):** When neither of the two types of steps defined above is deemed appropriate, the current iterate x_k satisfies the approximate optimality conditions of Definition 1, except for the condition (9b) on the scaled Hessian. We therefore check this condition and, if it is not satisfied, find a scaled negative curvature step that will lead to a significant decrease in f. While the other type of negative curvature step (obtained from Capped CG) changes only the apparently-free components, this scaled negative curvature step changes *all* components, in general. We believe that this type of step will rarely be taken; most instances of negative curvature will be detected during computation of the Newton-CG step.

Connections to known methods for bound-constrained and unconstrained optimization. The way in which Algorithm 1 combines Newton-CG steps with gradient projection steps is inspired in part by Moré and Toraldo [21], who use CG iterations applied to the Newton system to "explore" a face of the feasible orthant and gradient projection to move to a new face. However, [21] addresses only convex quadratic problems and has no complexity analysis.

There are obvious connections between Algorithm 1 and the Newton-CG methods for unconstrained nonconvex optimization described in [23] and [24]. The latter methods make use of Capped CG procedures (where the "cap" refers to an implicit bound on the number of CG iterations allowed at each invocation), as well as negative curvature directions and backtracking line searches. We leverage the similarities by using the same "subroutines" for Capped CG and negative curvature detection as in [23]; these methods are stated for completeness in Appendices B and C, along with their key properties. However, the modifications required to adapt the approach of [23] to handle bound constraints, in a way that allows complexity results to be proved, are significant and non-obvious. For one thing, we cannot simply project the approximate Newton step onto the feasible region, as this may not yield descent even for convex f; see [2, Section 1.5]. Indeed, Bertsekas proposed the two-metric gradient projection approach precisely to deal with this issue. Essentially, the proximity of iterates x_k to the boundary of the feasible set Ω and the use of projection inhibit steps in ways that may prevent the "significant decrease" in objective f required at each iteration to prove complexity. We need to use scaling of steps and Hessians, modified steplength acceptance criteria, and novel partitions of the set of components to overcome this potential hazard. Differ-

ences with prior work, particularly the unconstrained Newton-CG approach of [23], can be summarized as follows.

1. Our partition of $\{1, 2, \ldots, n\}$ into apparently-active and apparently-free parts (16) differs from standard two-metric gradient projection in not considering the sign of the gradient.
2. We use a gradient projection step in certain conditions; devising these conditions in such a way that the step yields the significant improvement in f required by our complexity analysis (see Lemma 2) is somewhat intricate.
3. We need a different sufficient decrease criterion for the Newton-CG step from the one in [23], and this step takes place only in the subspace of apparently-free variables. The analysis in proofs of Lemmas 3 and 4 is similar to that of corresponding results in [23], but takes the presence of bound constraints in the apparently-free variables into account.
4. We compute the full-dimensional negative curvature direction on a *diagonally scaled* version of the Hessian, and need a scaled direction and a different sufficient decrease condition from [23] (see Lemma 9).

4.2 Complexity of Algorithm 1

The following four results — Lemmas 2 to 5 — prove a lower bound on the amount of decrease in f at a single iteration in each of the following four cases. (We assume that Assumptions 1 and 2 hold with Ω in (2) for all these results, although we do not mention them in the statement of each result.)

(i) A gradient projection step is taken (Lemma 2);
(ii) The Newton-CG step is triggered and the Capped CG algorithm returns $d\_\text{type} = \text{NC}$, resulting in a negative curvature step involving the apparently-free components (Lemma 3);
(iii) The Newton-CG step is triggered and the Capped CG algorithm returns $d\_\text{type} = \text{SOL}$, resulting in a Newton-like step (Lemma 4);
(iv) The MEO procedure returns a negative curvature direction instead of a certificate of optimality, and a negative curvature step is taken (Lemma 5).

We state and prove these results without further elaboration.

Lemma 2 *Suppose that $J_k^+ \neq \emptyset$ at iteration k, and that $g_k^i < -\epsilon_k^{3/2}$ for some $i \in J_k^+$ or $\|S_k^+ g_k^+\| > \epsilon_k^2$, so that a projected gradient step is taken. Then*

$$f(x_k) - f(x_{k+1}) > \frac{1}{4} \min\{\theta/L_g, 1\}\epsilon_k^3.$$

Proof If $g_k^i < -\epsilon_k^{3/2}$ for some $i \in J_k^+$ or $\|S_k^+ g_k^+\| > \epsilon_k^2$ at the gradient projection step, then for any steplength $\beta > 0$, at least one of two cases occurs. In the first case of $g_k^i < -\epsilon_k^{3/2}$ for some $i \in J_k^+$, we have

$$g_k^i < -\epsilon_k^{3/2} \implies (g_k^i)^2 > \epsilon_k^3 \implies (x_k^i - (x_k^i - \beta g_k^i)_+)g_k^i = \beta(g_k^i)^2 > \beta\epsilon_k^3. \quad (18)$$

In the second case, we have

$$\|S_k^+ g_k^+\|^2 > \epsilon_k^4 \implies \sum_{i\in J_k^+} (x_k^i)^2 (g_k^i)^2 > \epsilon_k^4$$
$$\implies \sum_{i\in J_k^+, \beta g_k^i \le x_k^i} (x_k^i)^2 (g_k^i)^2 + \sum_{i\in J_k^+, \beta g_k^i > x_k^i} (x_k^i)^2 (g_k^i)^2 > \epsilon_k^4.$$

Therefore, either

$$\sum_{i\in J_k^+, \beta g_k^i \le x_k^i} (x_k^i)^2 (g_k^i)^2 \ge \epsilon_k^4/2 \overset{(x_k^i \le \epsilon_k, \forall i \in J_k^+)}{\implies} \sum_{i\in J_k^+, \beta g_k^i \le x_k^i} (g_k^i)^2 \ge \epsilon_k^2/2,$$

or

$$\sum_{i\in J_k^+, \beta g_k^i > x_k^i} (x_k^i)^2 (g_k^i)^2 \ge \epsilon_k^4/2 \implies \sum_{i\in J_k^+, \beta g_k^i > x_k^i} x_k^i g_k^i \ge \epsilon_k^2/\sqrt{2}.$$

Thus in this case, we have

$$\begin{aligned}\sum_{i\in J_k^+} (x_k^i - (x_k^i - \beta g_k^i)_+) g_k^i &= \sum_{i\in J_k^+, \beta g_k^i \le x_k^i} \beta (g_k^i)^2 + \sum_{i\in J_k^+, \beta g_k^i > x_k^i} x_k^i g_k^i \\ &\ge \min\{\beta/2, 1/\sqrt{2}\}\epsilon_k^2 \\ &\overset{(\epsilon_k<1)}{>} \min\{\beta/2, 1/\sqrt{2}\}\epsilon_k^3. \end{aligned} \tag{19}$$

By noting $g_k^i(x_k^i - (x_k^i - \beta g_k^i)_+) \ge 0$ for any $i \in \mathcal{I}$, we have for any $\beta > 0$ that

$$\begin{aligned} g_k^T(x_k - P(x_k - \beta g_k)) &= \sum_{i\in\mathcal{I}} g_k^i (x_k^i - (x_k^i - \beta g_k^i)_+) + \sum_{i\in\mathcal{I}^c} \beta (g_k^i)^2 \\ &\ge \sum_{i\in J_k^+} g_k^i (x_k^i - (x_k^i - \beta g_k^i)_+) \\ &\overset{(18),(19)}{>} \min\{\beta/2, 1/\sqrt{2}\}\epsilon_k^3. \end{aligned} \tag{20}$$

Note for any $0 < \beta < \frac{1}{L_g}$, where L_g is the Lipschitz constant of ∇f, we have

$$\begin{aligned} f(P(x_k - \beta g_k)) &\le f(x_k) - g_k^T(x_k - P(x_k - \beta g_k)) + \frac{L_g}{2}\|x_k - P(x_k - \beta g_k)\|^2 \\ &\le f(x_k) - g_k^T(x_k - P(x_k - \beta g_k)) + \frac{L_g}{2}\beta g_k^T(x_k - P(x_k - \beta g_k)) \\ &< f(x_k) - g_k^T(x_k - P(x_k - \beta g_k)) + \frac{1}{2} g_k^T(x_k - P(x_k - \beta g_k)) \\ &= f(x_k) - \frac{1}{2} g_k^T(x_k - P(x_k - \beta g_k)), \end{aligned}$$

where the second inequality holds because $(u - v)^T(P(u) - P(v)) \ge \|P(u) - P(v)\|^2$ for any $u, v \in \mathbb{R}^n$, and the third inequality holds because $\beta < 1/L_g$

and $g_k^T(x_k - P(x_k - \beta g_k)) > 0$ by (20). Therefore, by the line search rule, $\tilde{m}_k < +\infty$ and $\theta^{\tilde{m}_k} \geq \min\left\{\frac{\theta}{L_g}, 1\right\}$. Thus, by the lower bound for $\theta^{\tilde{m}_k}$, the bound (20), and the backtracking line search mechanism, we have

$$f(x_k) - f(x_{k+1}) > \frac{1}{2} g_k^T (x_k - P(x_k - \theta^{\tilde{m}_k} g_k)) > \frac{1}{4} \min\{\theta/L_g, 1\}\, \epsilon_k^3.$$

□

Lemma 3 *Suppose that at iteration k, a Newton-CG step is triggered and that Algorithm 3 returns $d\_\text{type} = \text{NC}$. Then we have $m_k < +\infty$ and*

$$f(x_k) - f(P(x_k + \alpha_k d_k)) > c_{\text{nc}} \epsilon_k^3,$$

where $c_{\text{nc}} \triangleq \eta \min\left\{\frac{(3-6\eta)^2\theta^2}{L_H^2}, \theta^2\right\}$.

Proof For the Newton-CG step, if $\|\alpha d_k\| \leq \epsilon_k$ for some $\alpha > 0$, then $\|\alpha d_k^-\|_\infty = \|\alpha d_k\|_\infty \leq \epsilon_k$ and $P(x_k + \alpha d_k) = x_k + \alpha d_k$. From (6c), we have

$$\begin{aligned} f(P(x_k + \alpha d_k)) &= f(x_k + \alpha d_k) \\ &\leq f(x_k) + \alpha g_k^T d_k + \frac{\alpha^2}{2} d_k^T H_k d_k + \frac{L_H}{6}\alpha^3 \|d_k\|^3. \end{aligned} \tag{21}$$

Since $d\_\text{type} = \text{NC}$, we have that $(d_k^-)^T g_k^- \leq 0$, and from Lemma 7(let $\bar{d} = d_k^-, \epsilon = \epsilon_k$) that $\frac{(d_k^-)^T H_k^- d_k^-}{\|d_k^-\|^2} = -\|d_k^-\| \leq -\epsilon_k$. Then for any $0 < \alpha < \frac{3-6\eta}{L_H}$,

$$\begin{aligned} &f(x_k) + \alpha g_k^T d_k + \frac{\alpha^2}{2} d_k^T H_k d_k + \frac{L_H}{6}\alpha^3 \|d_k\|^3 \\ &= f(x_k) + \alpha (g_k^-)^T d_k^- + \frac{\alpha^2}{2} (d_k^-)^T H_k^- d_k^- + \frac{L_H}{6}\alpha^3 \|d_k^-\|^3 \\ &\leq f(x_k) - \frac{\alpha^2}{2}\|d_k^-\|^3 + \frac{L_H}{6}\alpha^3 \|d_k^-\|^3 \\ &< f(x_k) - \eta\alpha^2 \|d_k^-\|^3 \leq f(x_k) - \eta\alpha^2 \epsilon_k \|d_k\|^2. \end{aligned} \tag{22}$$

Then, by leveraging (21) and (22), we have that if $\alpha < \min\left\{\frac{3-6\eta}{L_H}, \frac{\epsilon_k}{\|d_k\|}\right\}$, then $f(P(x_k + \alpha d_k)) < f(x_k) - \eta\alpha^2\epsilon_k\|d_k\|^2$. Therefore, backtracking will terminate when α_k drops below $\min\left\{\frac{3-6\eta}{L_H}, \frac{\epsilon_k}{\|d_k\|}\right\}$, if not earlier. Further, because of the backtracking mechanism, α_k cannot be less than θ times this value. As a result, we have

$$\begin{aligned} \alpha_k &\geq \min\left\{\theta \min\left\{\frac{3-6\eta}{L_H}, \frac{\epsilon_k}{\|d_k\|}\right\}, 1\right\} \\ \implies \alpha_k \|d_k\| &\geq \min\left\{\frac{(3-6\eta)\theta\|d_k\|}{L_H}, \theta\epsilon_k, \|d_k\|\right\} \\ &\overset{(\|d_k\| \geq \epsilon_k)}{\geq} \min\left\{\frac{(3-6\eta)\theta}{L_H}, \theta, 1\right\}\epsilon_k \end{aligned}$$

$$\implies \alpha_k^2 \epsilon_k \|d_k\|^2 \geq \min\left\{\frac{(3-6\eta)^2\theta^2}{L_H^2}, \theta^2\right\}\epsilon_k^3.$$

Also, $\|d_k\| = \|d_k^-\| = \frac{|(d_k^-)^T H_k^- d_k^-|}{\|d_k^-\|^2} \leq \|H_k^-\|_2 \leq \|H_k\|_2 \leq L_g$ and

$$\alpha_k \geq \min\left\{\theta \min\left\{\frac{3-6\eta}{L_H}, \frac{\epsilon_k}{\|d_k\|}\right\}, 1\right\} \overset{(\|d_k\| \leq L_g)}{\geq} \min\left\{\frac{(3-6\eta)\theta}{L_H}, \frac{\theta\epsilon_k}{L_g}, 1\right\}$$
$$\implies m_k = \log_\theta \alpha_k \leq \max\left\{\log_\theta\left(\frac{(3-6\eta)\theta}{L_H}\right), \log_\theta\left(\frac{\theta\epsilon_k}{L_g}\right), 0\right\},$$

verifying that m_k is finite and completing the proof. □

Lemma 4 *Suppose that at iteration k, a Newton-CG step is triggered. Moreover, Algorithm 3 returns $d\_\text{type} = \text{SOL}$. Then $m_k < +\infty$ and*

$$f(x_k) - f(P(x_k + \alpha_k d_k)) > c_{\text{sol}} \min\{\|\nabla f(P(x_k + \alpha_k d_k))\,|_{J_k^-}\,\|^2 \epsilon_k^{-1}, \epsilon_k^3\}, \quad (23)$$

where

$$c_{\text{sol}} \triangleq \eta \min\left\{\frac{4}{25 + 8L_H}, \theta^2, \frac{9(1-\zeta-2\eta)^2\theta^2}{L_H^2}, \frac{(1-\zeta)^2\theta^2}{(L_H/3 + 2\eta)^2}\right\}.$$

Proof Define

$l_k \triangleq \min\left\{l \in \mathbb{N} \mid \theta^l \|d_k\| \leq \epsilon_k\right\}$

$j_k \triangleq$

$$\min\left\{j \geq l_k, j \in \mathbb{N} \mid \theta^j g_k^T d_k + \frac{\theta^{2j}}{2} d_k^T H_k d_k + \frac{L_H \theta^{3j}}{6}\|d_k\|^3 < -\eta\theta^{2j}\epsilon_k\|d_k\|^2\right\}.$$

Then from (21) and the definition of j_k, we have that

$$f(P(x_k + \theta^{j_k} d_k)) < f(x_k) - \eta\theta^{2j_k}\epsilon_k\|d_k\|^2.$$

Therefore, by the definition of m_k in Algorithm 1, it follows that $m_k \leq j_k$. By Lemma 7($d = d_k^-$, $g = g_k^-$, $\epsilon = \epsilon_k$), we have

$$\|d_k\| = \|d_k^-\| \leq 1.1\epsilon_k^{-1}\|g_k^-\| \leq 1.1\epsilon_k^{-1}\|g_k\| \leq 1.1\epsilon_k^{-1} L_f,$$

so that

$$l_k \leq \left[\log_\theta\left(\frac{\epsilon_k}{\|d_k\|}\right)\right]_+ + 1 \leq \left[\log_\theta\left(\frac{\epsilon_k^2}{1.1 L_f}\right)\right]_+ + 1. \quad (24)$$

According to Lemma 7 (with $d = d_k^-$, $H = H_k^-$, $g = g_k^-$, $\epsilon = \epsilon_k$), we have that

$$(d_k^-)^T(H_k^- + 2\epsilon_k I)d_k^- \geq \epsilon_k\|d_k^-\|^2, \quad (25a)$$
$$\|r_k^-\| \leq \frac{1}{2}\epsilon_k\zeta\|d_k^-\|, \quad (25b)$$

where $r_k^- \triangleq (H_k^- + 2\epsilon_k I)d_k^- + g_k^-$. Then,

$$\begin{aligned}
&\theta^j (g_k^-)^T d_k^- + \frac{\theta^{2j}}{2}(d_k^-)^T H_k^- d_k^- + \frac{L_H \theta^{3j}}{6}\|d_k^-\|^3 \\
&= -\theta^j (H_k^- d_k^- + 2\epsilon_k d_k^- - r_k^-)^T d_k^- + \frac{\theta^{2j}}{2}(d_k^-)^T H_k^- d_k^- + \frac{L_H \theta^{3j}}{6}\|d_k^-\|^3 \\
&= -\theta^j \left(1 - \frac{\theta^j}{2}\right)(d_k^-)^T (H_k^- + 2\epsilon_k I) d_k^- - \epsilon_k \theta^{2j}\|d_k^-\|^2 - \theta^j (r_k^-)^T d_k^- \\
&+ \frac{L_H \theta^{3j}}{6}\|d_k^-\|^3 \\
&\overset{(25a)}{\leq} -\theta^j \left(1 - \frac{\theta^j}{2}\right)\epsilon_k \|d_k^-\|^2 + \theta^j \|r_k^-\|\|d_k^-\| + \frac{L_H \theta^{3j}}{6}\|d_k^-\|^3 \\
&\overset{(25b)}{\leq} -\frac{\theta^j}{2}\epsilon_k \|d_k^-\|^2 + \frac{\theta^j}{2}\epsilon_k \zeta \|d_k^-\|^2 + \frac{L_H \theta^{3j}}{6}\|d_k^-\|^3 \\
&= -\frac{\theta^j}{2}(1-\zeta)\epsilon_k \|d_k^-\|^2 + \frac{L_H \theta^{3j}}{6}\|d_k^-\|^3. \qquad (26)
\end{aligned}$$

It can be verified that for any $j \geq \left[\log_\theta \left(\frac{(1-\zeta)\epsilon_k}{\eta\epsilon_k + \sqrt{\eta^2\epsilon_k^2 + 1.1 L_H (1-\zeta) L_f/3}}\right)\right]_+ + 1$, we have

$$\begin{aligned}
\theta^j &< \frac{(1-\zeta)\epsilon_k}{\eta\epsilon_k + \sqrt{\eta^2\epsilon_k^2 + 1.1 L_H (1-\zeta) L_f/3}} \\
\overset{(\|d_k^-\| \leq 1.1\epsilon_k^{-1} L_f)}{\Longrightarrow} \theta^j &< \frac{(1-\zeta)\epsilon_k}{\eta\epsilon_k + \sqrt{\eta^2\epsilon_k^2 + L_H (1-\zeta)\epsilon_k \|d_k^-\|/3}}.
\end{aligned}$$

It then follows from the quadratic formula applied to the following quadratic in θ^j [2]

$$\begin{aligned}
&\frac{L_H\|d_k^-\|}{6}\theta^{2j} + \eta\epsilon_k\theta^j - \frac{(1-\zeta)\epsilon_k}{2} && < 0 \\
\Longrightarrow\ &-\frac{\theta^j}{2}(1-\zeta)\epsilon_k\|d_k^-\|^2 + \frac{L_H}{6}\theta^{3j}\|d_k^-\|^3 && < -\eta\theta^{2j}\epsilon_k\|d_k^-\|^2 \\
\overset{(26)}{\Longrightarrow}\ &\theta^j (g_k^-)^T d_k^- + \frac{\theta^{2j}}{2}(d_k^-)^T H_k^- d_k^- + \frac{L_H\theta^{3j}}{6}\|d_k^-\|^3 && < -\eta\theta^{2j}\epsilon_k\|d_k^-\|^2 \\
\Longrightarrow\ &\theta^j g_k^T d_k + \frac{\theta^{2j}}{2} d_k^T H_k d_k + \frac{L_H\theta^{3j}}{6}\|d_k\|^3 && < -\eta\theta^{2j}\epsilon_k\|d_k\|^2.
\end{aligned}$$

Then by the definitions of j_k and l_k together with (24), we have

$$\begin{aligned}
&j_k \leq 1 + \\
&\max\left\{\left[\log_\theta\left(\frac{\epsilon_k^2}{1.1 L_f}\right)\right]_+, \left[\log_\theta\left(\frac{(1-\zeta)\epsilon_k}{\eta\epsilon_k + \sqrt{\eta^2\epsilon_k^2 + 1.1 L_H (1-\zeta) L_f/3}}\right)\right]_+\right\},
\end{aligned}$$

[2] $z \geq 0$ and $az^2 + bz + c < 0$ together are equivalent to $0 \leq z < \frac{-b+\sqrt{b^2-4ac}}{2a} = \frac{-2c}{b+\sqrt{b^2-4ac}}$.

which is also an upper bound for m_k.

Next, we derive the lower bound for $\alpha_k^2 \epsilon_k \|d_k\|^2$ which, when scaled by η, is the required amount of decrease in f. We consider four cases.

Case 1. $j_k = l_k = 0$. In this case we have $m_k = 0$, $\alpha_k = 1$, and $\|d_k^-\| = \|d_k\| \leq \epsilon_k$. Therefore, $x_k^i + d_k^i \geq 0, \forall i \in \mathcal{I} \cap J_k^- \implies P(x_k + \alpha_k d_k) = x_k + d_k$. Then we have

$$\begin{aligned}
\|\nabla f(P(x_k + \alpha_k d_k)) \mid_{J_k^-} \| &= \|\nabla f(x_k + d_k) \mid_{J_k^-} \| \\
&= \|\nabla f(x_k + d_k) \mid_{J_k^-} - g_k^- + g_k^- \| \\
&= \|\nabla f(x_k + d_k) \mid_{J_k^-} - g_k^- - H_k^- d_k^- - 2\epsilon_k d_k^- + r_k^- \| \\
&\leq \frac{L_H}{2} \|d_k^-\|^2 + 2\epsilon_k \|d_k^-\| + \|r_k^-\| \\
&\overset{(25b)}{\leq} \frac{L_H}{2} \|d_k^-\|^2 + \frac{4+\zeta}{2} \epsilon_k \|d_k^-\| \\
&\overset{(\zeta<1)}{\leq} \frac{L_H}{2} \|d_k^-\|^2 + \frac{5}{2} \epsilon_k \|d_k^-\|.
\end{aligned}$$

By applying the quadratic formula to the inequality above (which involves a quadratic in $\|d_k^-\|$), we obtain

$$\begin{aligned}
\|d_k^-\| &\geq \frac{-\frac{5}{2} + \sqrt{\frac{25}{4} + 2L_H \|\nabla f(P(x_k + \alpha_k d_k)) \mid_{J_k^-} \| / \epsilon_k^2}}{L_H} \cdot \epsilon_k \\
&= \frac{-5 + \sqrt{25 + 8L_H \min\{\|\nabla f(P(x_k + \alpha_k d_k)) \mid_{J_k^-} \| / \epsilon_k^2, 1\}}}{2L_H} \cdot \epsilon_k \\
&= \frac{4 \min\{\|\nabla f(P(x_k + \alpha_k d_k)) \mid_{J_k^-} \| / \epsilon_k^2, 1\}}{5 + \sqrt{25 + 8L_H \min\{\|\nabla f(P(x_k + \alpha_k d_k)) \mid_{J_k^-} \| / \epsilon_k^2, 1\}}} \cdot \epsilon_k \\
&\geq \frac{4}{5 + \sqrt{25 + 8L_H}} \min\{\|\nabla f(P(x_k + \alpha_k d_k)) \mid_{J_k^-} \| \epsilon_k^{-1}, \epsilon_k\} \\
&\geq \frac{2}{\sqrt{25 + 8L_H}} \min\{\|\nabla f(P(x_k + \alpha_k d_k)) \mid_{J_k^-} \| \epsilon_k^{-1}, \epsilon_k\} \\
&\Downarrow (\alpha_k = 1, \|d_k\| = \|d_k^-\|) \\
\alpha_k^2 \epsilon_k \|d_k\|^2 &\geq \frac{4}{25 + 8L_H} \min\{\|\nabla f(P(x_k + \alpha_k d_k)) \mid_{J_k^-} \|^2 \epsilon_k^{-1}, \epsilon_k^3\}.
\end{aligned}$$

Case 2. $j_k = l_k \geq 1$. In this case, since $\alpha_k = \theta^{m_k}$ with $m_k \leq j_k = l_k$, we have

$$\begin{aligned}
\theta^{l_k} \|d_k\| > \theta \epsilon_k &\implies \alpha_k \|d_k\| = \theta^{m_k} \|d_k\| > \theta \epsilon_k \\
&\implies \alpha_k^2 \epsilon_k \|d_k\|^2 = (\alpha_k \|d_k\|)^2 \epsilon_k > \theta^2 \epsilon_k^3.
\end{aligned}$$

Case 3. $j_k > l_k = 0$. For $j = 0$ and $j = j_k - 1$, we must have

$$\theta^j g_k^T d_k + \frac{\theta^{2j}}{2} d_k^T H_k d_k + \frac{L_H \theta^{3j}}{6} \|d_k\|^3 \geq -\eta \theta^{2j} \epsilon_k \|d_k\|^2$$

$$\begin{aligned}
&\Longrightarrow \theta^j (g_k^-)^T d_k^- + \frac{\theta^{2j}}{2}(d_k^-)^T H_k^- d_k^- + \frac{L_H \theta^{3j}}{6}\|d_k^-\|^3 \geq -\eta\theta^{2j}\epsilon_k\|d_k^-\|^2 \\
&\overset{(26)}{\Longrightarrow} -\frac{\theta^j}{2}(1-\zeta)\epsilon_k\|d_k^-\|^2 + \frac{L_H}{6}\theta^{3j}\|d_k^-\|^3 \geq -\eta\theta^{2j}\epsilon_k\|d_k^-\|^2 \\
&\Longrightarrow \frac{L_H}{6}\theta^{2j} + \frac{\eta\epsilon_k}{\|d_k^-\|}\theta^j - \frac{(1-\zeta)\epsilon_k}{2\|d_k^-\|} \geq 0. \qquad (27)
\end{aligned}$$

By setting $j = 0$ in this inequality, we have $\|d_k^-\| \geq (3(1-\zeta) - 6\eta)\epsilon_k / L_H$. By setting $j = j_k - 1$ in this same inequality, and using $\theta^{j_k} > \theta^{2j_k}$, we have

$$\begin{aligned}
\left(\frac{L_H}{6} + \frac{\eta\epsilon_k}{\|d_k^-\|}\right)\theta^{j_k - 1} &\geq \frac{(1-\zeta)\epsilon_k}{2\|d_k^-\|} \\
\Longrightarrow\ \theta^{j_k}\|d_k^-\| &\geq \frac{(1-\zeta)\theta\epsilon_k}{(L_H/3) + 2\eta\epsilon_k/\|d_k^-\|} \qquad (28) \\
&\geq \frac{(1-\zeta)\theta\epsilon_k}{(L_H/3) + 2\eta L_H/(3(1-\zeta) - 6\eta)} \\
&= \frac{3(1-\zeta-2\eta)\theta\epsilon_k}{L_H},
\end{aligned}$$

where the final equality follows by elementary manipulation. Using again $\alpha_k = \theta^{m_k} \geq \theta^{j_k}$, we have

$$\alpha_k^2\epsilon_k\|d_k\|^2 = \alpha_k^2\epsilon_k\|d_k^-\|^2 \geq (\theta^{j_k}\|d_k^-\|)^2\epsilon_k \geq \frac{9(1-\zeta-2\eta)^2\theta^2\epsilon_k^3}{L_H^2}.$$

Case 4. $j_k > l_k \geq 1$. By the same argument as in Case 3, (28) holds. Moreover, $\|d_k^-\| = \|d_k\| > \epsilon_k$ since $l_k \geq 1$. Therefore, we have

$$\begin{aligned}
(28) \ &\Longrightarrow\ \theta^{j_k}\|d_k^-\| \geq \frac{(1-\zeta)\theta\epsilon_k}{L_H/3 + 2\eta\epsilon_k/\|d_k^-\|} > \frac{(1-\zeta)\theta\epsilon_k}{L_H/3 + 2\eta} \\
&\Longrightarrow\ \alpha_k^2\epsilon_k\|d_k\|^2 \geq (\theta^{j_k}\|d_k\|)^2\epsilon_k \geq \frac{(1-\zeta)^2\theta^2\epsilon_k^3}{(L_H/3 + 2\eta)^2}
\end{aligned}$$

By combining the four cases analyzed above, we obtain

$$\alpha_k^2\epsilon_k\|d_k\|^2 \geq \frac{1}{\eta}c_{\text{sol}}\min\{\|\nabla f(P(x_k + \alpha_k d_k))\,|_{J_k^-}\,\|^2\epsilon_k^{-1}, \epsilon_k^3\}.$$

Therefore, by the line search rule, (23) holds. □

Lemma 5 *Suppose that at iteration k of Algorithm 1, Procedure 4 is invoked and identifies a direction with curvature less than or equal to $-\frac{1}{2}\epsilon_k$. Then we have*

$$f(x_k) - f(x_{k+1}) > \eta\min\left\{\frac{(3-6\eta)^2\theta^2}{8L_H^2}, \frac{\theta^2}{2}, \frac{1}{8}\right\}\epsilon_k^3 \geq \min\left\{\frac{c_{\text{nc}}}{8}, \frac{\eta}{8}\right\}\epsilon_k^3.$$

Proof Let scalar λ and vector d be the quantities returned by MEO, Procedure 4, so that $d^T S_k H_k S_k d = \lambda \leq -\epsilon_k/2$ and $\|d\| = 1$. From the subsequent definition of d_k in Algorithm 1, we have that

$$g_k^T S_k d_k = -|g_k^T S_k d||d^T S_k H_k S_k d| \leq 0, \tag{29a}$$

$$\|d_k\| = |d^T S_k H_k S_k d|\|d\| = |\lambda| \geq \tfrac{1}{2}\epsilon_k, \tag{29b}$$

$$d_k S_k H_k S_k d_k = (d^T S_k H_k S_k d)^3 = \lambda^3 = -\|d_k\|^3. \tag{29c}$$

Then, for any $0 < \gamma < \frac{3-6\eta}{L_H}$, we have

$$\begin{aligned} f(x_k + \gamma S_k d_k) &\leq f(x_k) + \gamma g_k^T S_k d_k + \frac{\gamma^2}{2} d_k^T S_k H_k S_k d_k + \frac{L_H}{6}\gamma^3 \|S_k d_k\|^3 \\ &\overset{(S_k[i,i]\leq 1)}{\leq} f(x_k) + \gamma g_k^T S_k d_k + \frac{\gamma^2}{2} d_k^T S_k H_k S_k d_k + \frac{L_H}{6}\gamma^3 \|d_k\|^3 \\ &\overset{(29)}{\leq} f(x_k) - \frac{\gamma^2}{2}\|d_k\|^3 + \frac{L_H}{6}\gamma^3\|d_k\|^3 \\ &< f(x_k) - \eta\gamma^2\|d_k\|^3. \end{aligned}$$

Note that if $\gamma\|d_k\| \leq \epsilon_k < 1$ then $\gamma\|d_k\|_\infty \leq \epsilon_k < 1$ and $P(x_k + \gamma S_k d_k) = x_k + \gamma S_k d_k$. In fact, by invoking (17), we have

$$\begin{aligned} i \in J_k^+ &\implies x_k^i + \gamma s_k^i d_k^i \geq x_k^i - x_k^i \|\gamma d_k\|_\infty \geq 0, \\ i \in J_k^- \cap \mathcal{I} &\implies x_k^i + \gamma s_k^i d_k^i = x_k^i + \gamma d_k^i \geq x_k^i - \epsilon_k > 0. \end{aligned}$$

Thus for any $\gamma < \min\left\{\frac{3-6\eta}{L_H}, \frac{\epsilon_k}{\|d_k\|}\right\}$, we have

$$f(P(x_k + \gamma S_k d_k)) = f(x_k + \gamma S_k d_k) < f(x_k) - \eta\gamma^2\|d_k\|^3.$$

Therefore, because of the backtracking mechanism and the definition of $\bar{m}_k$, we have

$$\begin{aligned} \theta^{\bar{m}_k} &\geq \min\left\{\theta \min\left\{\frac{3-6\eta}{L_H}, \frac{\epsilon_k}{\|d_k\|}\right\}, 1\right\} \\ \implies \theta^{\bar{m}_k}\|d_k\| &\geq \min\left\{\frac{(3-6\eta)\theta\|d_k\|}{L_H}, \theta\epsilon_k, \|d_k\|\right\} \\ &\overset{(\|d_k\|=|\lambda|\geq\frac{\epsilon_k}{2})}{\geq} \min\left\{\frac{(3-6\eta)\theta}{2L_H}, \theta, \frac{1}{2}\right\}\epsilon_k. \end{aligned} \tag{30}$$

Then, based on the line search rule and the bounds (30) and (29b), we have

$$\begin{aligned} f(x_k) - f(x_{k+1}) &= f(x_k) - f(P(x_k + \theta^{\bar{m}_k} S_k d_k)) \\ &> \eta\theta^{2\bar{m}_k}\|d_k\|^3 \\ &\geq \eta \min\left\{\frac{(3-6\eta)^2\theta^2}{8L_H^2}, \frac{\theta^2}{2}, \frac{1}{8}\right\}\epsilon_k^3. \end{aligned}$$

The final inequality follows from the definition of $c_{\rm nc}$ in Lemma 3. □

We now state and prove the main complexity result for Algorithm 1. Note that ϵ_g is the parameter in the condition triggering the Newton-CG step in Algorithm 1.

Theorem 3 *Suppose that Assumptions 1 and 2 hold for the problem* (1), (2). *Consider Algorithm 1 with* $\epsilon_k \equiv \epsilon_H < 1$. *Then Algorithm 1 will stop within*

$$K_{\text{pncg}} \triangleq \left\lfloor \frac{16(f(x_0) - f_{\text{low}})}{\min\left\{c_{\text{nc}}, 8c_{\text{sol}}, \frac{2\theta}{L_g}, \eta\right\}} \max\{\epsilon_g^{-2}\epsilon_H, \epsilon_H^{-3}\} \right\rfloor + 2 \tag{31}$$

iterations, and outputs a vector $x \in \Omega$ *such that the following approximate first-order optimality conditions hold*

$$x^i \geq 0 \ \text{ for } i \in \mathcal{I}, \quad \|S\nabla f(x)\| \leq \epsilon_g + \epsilon_H^2, \tag{32a}$$

$$\nabla_i f(x) \geq -\epsilon_H^{3/2}, \quad \forall i \in J^+ \triangleq \{i \in \mathcal{I} \mid 0 \leq x^i \leq \epsilon_H\}, \tag{32b}$$

with probability 1. Moreover, $S\nabla^2 f(x)S \succeq -\epsilon_H I$ *with probability at least* $(1-\delta)^{K_{\text{pncg}}}$, *where* $S = \text{diag}(s)$ *is a diagonal matrix with* $s^i = x^i, \forall i \in J^+$ *and* $s^i = 1$ *otherwise; and* $\delta \in [0, 1)$ *is the probability of failure in Procedure 4. In particular, if we set* $\epsilon_g = \epsilon$ *and* $\epsilon_H = \sqrt{\epsilon}$, *then the algorithm outputs an* $(\epsilon, 1/2)$*-2o point (according to Definition 1) with probability at least* $(1-\delta)^{K_{\text{pncg}}}$ *within* $\mathcal{O}(\epsilon^{-3/2})$ *iterations.*

Proof We prove by estimating the function decrease when the algorithm does not stop at iteration k or $k+1$.

Case 1. A gradient projection step is taken at iteration k. Then by Lemma 2, we have

$$f(x_k) - f(x_{k+1}) > \frac{1}{4}\min\left\{\frac{\theta}{L_g}, 1\right\}\epsilon_k^3. \tag{33}$$

Case 2. The Newton-CG step is triggered at iteration k, $J_{k+1}^- \neq \emptyset$ and $\|g_{k+1}^-\| > \epsilon_g$. Note that $\epsilon_k \equiv \epsilon_H$ indicates that $J_{k+1}^- \subseteq J_k^-$. Therefore, we have

$$\|\nabla f(x_{k+1}) \mid_{J_k^-} \| \geq \|g_{k+1}^-\| > \epsilon_g.$$

Thus, by Lemma 3 and Lemma 4, we have that

$$\begin{aligned} f(x_k) - f(x_{k+1}) &\geq \min\{c_{\text{nc}}, c_{\text{sol}}\}\min\{\|\nabla f(x_{k+1}) \mid_{J_k^-} \|^2\epsilon_H^{-1}, \epsilon_H^3\} \\ &> \min\{c_{\text{nc}}, c_{\text{sol}}\}\min\{\epsilon_g^2\epsilon_H^{-1}, \epsilon_H^3\}. \end{aligned}$$

Case 3. The MEO procedure is triggered and a negative curvature step is taken at iteration k. Lemma 5 then implies that

$$f(x_k) - f(x_{k+1}) > \min\left\{\frac{c_{\text{nc}}}{8}, \frac{\eta}{8}\right\}\epsilon_k^3. \tag{34}$$

Case 4. The Newton-CG step is triggered at iteration k, but $J_{k+1}^- = \emptyset$ or $\|g_{k+1}^-\| \leq \epsilon_g$. We have from Lemmas 3 and 4 that $f(x_k) > f(x_{k+1})$. Moreover,

since the algorithm does not stop at iteration $k+1$, x_{k+2} is calculated from a step that is analyzed in either Case 1 or Case 3. It follows that either (33) or (34) is satisfied with k replaced by $k+1$.

We now combine the lower bounds for function value decrease derived in the above four cases, let $\epsilon_k \equiv \epsilon_H < 1$, and we have that for any $k \geq 0$ such that the algorithm does not stop at iteration k and $k+1$, that

$$f(x_k) - f(x_{k+2}) > \min\left\{c_{\text{sol}}, \frac{c_{\text{nc}}}{8}, \frac{\theta}{4L_g}, \frac{\eta}{8}\right\} \min\{\epsilon_g^2 \epsilon_H^{-1}, \epsilon_H^3\}$$

if the stopping criterion is not satisfied. Therefore, the algorithm must stop within the number of iterations stated in the theorem. When the algorithm stops, the output x_k satisfies:

$$\|g_k^-\| \leq \epsilon_g, \quad g_k^i \geq -\epsilon_H^{3/2}, \ \forall i \in J_k^+, \quad \|S_k^+ g_k^+\| \leq \epsilon_H^2. \tag{35}$$

Now let us derive the probability that the output x_k does not satisfies $S_k H_k S_k \succeq -\epsilon_H I$. Denote by $p_{k,F}$ the probability that the algorithm does not stop before iteration $k-1$ and x_k does not satisfy $\lambda_{\min}(S_k \nabla^2 f(x_k) S_k) \geq -\epsilon_H$. (We set $p_{0,F} \triangleq 1$.) Denote by $p_{k,F,stop}$ the probability that the algorithm stops at iteration k but x_k does not satisfy $\lambda_{\min}(S_k \nabla^2 f(x_k) S_k) \geq -\epsilon_H$. Therefore, since the failure probability of Procedure 4 is δ, we have that

$$p_{k,F,stop} \leq \delta p_{k,F}.$$

We know that the algorithm must stop within K_{pncg} number of iterations. Therefore, if we denote the probability of failure of PNCG as p_F, then

$$p_F = \sum_{k=0}^{K_{\text{pncg}}-1} p_{k,F,stop}.$$

We have that for any $k = 0, 1, \ldots, K_{\text{pncg}} - 1$ that

$$p_{k,F} + \sum_{t=0}^{k-1} p_{t,F,stop} \leq 1,$$

so that

$$p_{k,F,stop} \leq \delta \left(1 - \sum_{t=0}^{k-1} p_{t,F,stop}\right), \quad k = 0, 1, \ldots, K_{\text{pncg}} - 1.$$

Next we show that $\sum_{t=0}^{k} p_{t,F,stop} \leq 1 - (1-\delta)^{k+1}$, $k = 0, 1, \ldots, K_{\text{pncg}} - 1$ by induction. The claim is trivial for $k = 0$. Supposing that it holds when $k = \bar{k} \in \{0, 1, \ldots, K_{\text{pncg}} - 2\}$, we have

$$\sum_{t=0}^{\bar{k}+1} p_{t,F,stop} = \sum_{t=0}^{\bar{k}} p_{t,F,stop} + p_{\bar{k}+1,F,stop}$$

$$
\begin{aligned}
&\leq \sum_{t=0}^{\bar{k}} p_{t,F,stop} + \delta \left(1 - \sum_{t=0}^{\bar{k}} p_{t,F,stop}\right) \\
&= \delta + (1-\delta) \sum_{t=0}^{\bar{k}} p_{t,F,stop} \\
&\leq \delta + (1-\delta)[1 - (1-\delta)^{\bar{k}+1}] \\
&= 1 - (1-\delta)^{\bar{k}+2}.
\end{aligned}
$$

This proves that the desired bound holds for $k = \bar{k}+1$, completing the induction. Therefore, we have that

$$
p_F = \sum_{k=0}^{K_{\text{pncg}}-1} p_{k,F,stop} \leq 1 - (1-\delta)^{K_{\text{pncg}}}.
$$

Then we proved that with probability at least $(1-\delta)^{K_{\text{pncg}}}$, the output x_k satisfies $S_k H_k S_k \succeq -\epsilon_H I$. This condition for x_k combined with (35) indicate the output property. □

In the statement of Theorem 3, δ is a user-defined parameter. It can be chosen small enough to ensure that $(1-\delta)^{K_{\text{pncg}}}$ is large. Specifically, by Bernoulli's inequality, for $\delta \in [0,1)$ and $K \geq 1$,

$$
(1-\delta)^K \geq 1 - K\delta.
$$

If, for example, we set $\delta = 0.01/K_{\text{pncg}}$, then $(1-\delta)^{K_{\text{pncg}}} \geq 1 - 0.01 = 0.99$. Note that the value of δ only affects the operation complexity (involving Hessian-vectors products), which depends only *logarithmically* on δ (see Corollary 1 below). Therefore, we are free to choose very small values of δ without affecting the operation complexity significantly.

We now state a result for operation complexity of this approach, based on the fundamental operations of gradient evaluation and Hessian-vector products.

Corollary 1 *Suppose that Assumptions 1, 2 hold for the problem* (1), (2). *For some* $\epsilon \in (0,1)$, *consider Algorithm 1 with* $\epsilon_k \equiv \sqrt{\epsilon}$ *and* $\epsilon_g = \epsilon$. *Then Algorithm 1 stops and outputs an* $(\epsilon, 1/2)$*-2o point with probability at least* $(1-\delta)^{K_{\text{pncg}}}$ *(*K_{pncg} *defined in* (31)*) within*

$$
O\left(\epsilon^{-3/2} \min\left\{n, \epsilon^{-1/4} \log\left(\frac{n}{\delta\epsilon}\right)\right\}\right).
$$

fundamental operations (gradient evaluations or Hessian-vector products).

Proof The bound on Hessian-vector products before Algorithm 1 stops is:

$$\sum_{k=0}^{K_{\text{pncg}}-1} (\max\{2\min\{n, \mathbb{J}_k\} + 1, N_k^{\text{meo}}\}), \tag{36}$$

where $2\min\{n, \mathbb{J}_k\} + 1$ and N_k^{meo} are the bound on Hessian-vector products of the Capped CG and MEO procedure, respectively, at iteration k. By Lemma 8 and 9 in Appendix B and C, given $\kappa \triangleq \frac{\|H_k^-\|+\epsilon_k}{\epsilon_k} \leq \frac{L_g+\epsilon_k}{\epsilon_k}$, $\mathcal{C}_k^{\text{meo}} = \log\left(\frac{2.75n}{\delta^2}\right)\frac{\sqrt{\|H\|}}{2} \leq \log\left(2.75n/\delta^2\right)\sqrt{L_g}/2$ and $\epsilon_k \equiv \sqrt{\epsilon}$, we have that:

$$\begin{aligned}
\mathbb{J}_k &\leq \min\left\{n, \left\lceil\left(\sqrt{\kappa}+\frac{1}{2}\right)\log\left(\frac{144(\sqrt{\kappa}+1)^2\kappa^6}{\zeta^2}\right)\right\rceil\right\} \\
\implies \mathbb{J}_k &= \mathcal{O}\left(\min\left\{n, \epsilon^{-\frac{1}{4}}\log\left(\epsilon^{-1}\right)\right\}\right) \\
N_k^{\text{meo}} &= \min\left\{n, 1+\lceil\mathcal{C}_k^{\text{meo}}\epsilon_k^{-\frac{1}{2}}\rceil\right\} = \mathcal{O}\left(\min\left\{n, \epsilon^{-\frac{1}{4}}\log\left(n/\delta\right)\right\}\right),
\end{aligned}$$

Therefore, by Theorem 3 we have that

$$\begin{aligned}
(36) &\leq \sum_{k=0}^{K_{\text{pncg}}-1} 2(\max\{\mathbb{J}_k, N_k^{\text{meo}}\}+1) \\
&= \mathcal{O}\left(K_{\text{pncg}}\min\left\{n, \epsilon^{-\frac{1}{4}}\max\left\{\log\left(\epsilon^{-1}\right), \log\left(n/\delta\right)\right\}\right\}\right) \\
&= \mathcal{O}\left(K_{\text{pncg}}\min\left\{n, \epsilon^{-\frac{1}{4}}\left(\log\left(\epsilon^{-1}\right)+\log\left(n/\delta\right)\right)\right\}\right) \\
&= \mathcal{O}\left(\epsilon^{-\frac{3}{2}}\min\left\{n, \epsilon^{-\frac{1}{4}}\log\left(\frac{n}{\delta\epsilon}\right)\right\}\right)
\end{aligned}$$

Then the result follows by noticing that the number of gradient evaluation is bounded by the number of outer-loop iterations of Algorithm 1, i.e., K_{pncg}. □

5 Numerical experiment

We test the practicality of **PNCG** (Algorithm 1) by comparing it with several other approaches on the well-known Nonnegative Matrix Factorization (NMF) problem. The competitors include the gradient projection method (**pgrad**) described in [3, Section 3.3] (see Algorithm 2), a log-barrier Newton-CG (**LBNCG**) proposed in [22] for optimization with bounds, and two approaches that are specialized to NMF. Preliminary results show that **PNCG** contends well with **pgrad** and **LBNCG**, and is competitive with the specialized methods on problems with relatively low dimensions.[3] We use $\langle A, B\rangle$ to denote

[3] Experiments in this section are conducted using `Matlab R2018b` on MacBook Air 1.3 GHz Intel Core i5. Source codes of experiments in this section can be found at: `https://github.com/yue-xie/ProjectedNewton`.

the inner product of matrices $A, B \in \mathbb{R}^{d_1 \times d_2}$ defined by $Tr(A^T B)$, while the Frobenius norm is $\|A\|_F = \sqrt{\langle A, A \rangle}$.

NMF is stated as follows, for a given matrix $V \in \mathbb{R}^{m \times n}$:

$$\min_{W \in \mathbb{R}^{m \times r}, Y \in \mathbb{R}^{r \times n}} F(W, Y) \triangleq \frac{1}{2}\|WY - V\|_F^2, \quad \text{subject to } W \geq 0,\ Y \geq 0, \tag{37}$$

where the nonnegativity constraints apply componentwise, that is, all elements of W and Y are required to be nonnegative. NMF has a wide range of applications in image processing and text mining; see [13] for a comprehensive review.

In all following experiments, we create synthetic datasets following the approach in [18]: Matrices $\bar{W} \in \mathbb{R}^{m \times r}$ and $\bar{Y} \in \mathbb{R}^{r \times n}$ are generated randomly where each element has half standard normal distribution (to ensure $\bar{W} \geq 0$ and $\bar{Y} \geq 0$). Then approximately 60% of the elements of these matrices (chosen uniformly at random) are replaced by zeros. We then set $V = \bar{W}\bar{Y} + E$, where E is elementwise Gaussian with mean 0 and standard deviation of 5% of average elementwise magnitude of $\bar{W}\bar{Y}$. Finally, V is normalized such that its average elementwise magnitude is 1.

5.1 Comparison with other solvers with complexity guarantees

In this subsection we solve NMF using **PNCG** and other solvers, including the gradient projection method (**pgrad**) and the log-barrier Newton-CG (**LBNCG**). The former is a known practical method for constrained nonlinear optimization [3, Section 3.3]. However, it is only guaranteed to seek an approximate first-order optimal point; its complexity guarantees ($\mathcal{O}(\epsilon^{-2})$) (c.f. [12]) are generally worse than second-order methods ($\mathcal{O}(\epsilon^{-3/2})$) in the nonconvex regime. The latter is proposed in [22], which does have competitive complexity guarantees (see Table 1). Although **PNCG** and **LBNCG** are able to locate approximate second-order optimal solutions, we stop these algorithms as long as a first-order point is found or time/iteration limit is reached, so that comparison with **pgrad** is fair.

Methods. First we specify the methods implemented in the experiment and their settings. We make use here of notation ∇^P introduced in [19] and defined as follows:

$$\nabla_i^P f(x) = \begin{cases} \nabla_i f(x) & \text{if } x^i > 0 \text{ or } i \in \mathcal{I}^c, \\ \min\{0, \nabla_i f(x)\} & \text{if } x^i = 0 \text{ and } i \in \mathcal{I}. \end{cases} \tag{38}$$

(Note that $\nabla^P f(x) = 0$ implies the first-order optimality conditions of (7).)

1. **PNCG** (Algorithm 1)[4]: Set $\epsilon_g = 10^{-6}$, $\epsilon_k \equiv \sqrt{\epsilon_g}$, $\theta = \zeta = 1/2$, $\eta = 0.2$. For the parameter $\hat{\zeta}$ in Algorithm 3, we set it initially .1, but decrease by

[4] Note that Assumption 1 may not hold for (37), but we can modify the formulation to ensure this property, for example by adding elementwise upper bounds to W and H or adding a penalty $\|W^T W - YY^T\|^2$ to the objective. We omit these modifications to allow a more direct comparison with the specialized solvers for (37) described later.

a factor of 10 whenever the line search procedure in the outer-loop fails to find a descent direction, until a lower bound of $\frac{\zeta}{3\kappa}$ is reached. We do not use Procedure MEO, terminating Algorithm 1 when $g_k^i \geq -\epsilon_k^{3/2}$ for all $i \in J_k^+$ and $\|S_k^+ g_k^+\| \leq \epsilon_k^2$ and $\|g_k^-\| \leq \epsilon_g$, because we are interested only in finding an approximate first-order solution satisfying (32).

2. **pgrad** (Algorithm 2): Projected gradient method [3, Section 3.3] directly applied to NMF. This method uses Armijo rule along the projection arc [3], with backtracking parameter $\beta = 1/2$ and step acceptance parameter $\sigma = 1/2$ is chosen as such to be consistent with the gradient projection step in Algorithm 1 (where the step acceptance parameter is set as the default value 1/2). This algorithm is terminated when $\|\nabla^P F(W,Y)\|_F \leq 10^{-4}$.

Algorithm 2 Projected gradient method for NMF (**pgrad**)

(Initialization) Choose initial nonnegative real matrices W_0, Y_0, backtracking parameter $\beta \in (0,1)$ and step acceptance parameter $\sigma \in (0,1)$.
for $k = 0, 1, 2, \ldots$ **do**
Let m_k be the smallest nonnegative integer m such that

$$F(W_k, Y_k) - F(W_k(\beta^m), Y_k(\beta^m)) > \sigma(\langle W_k - W_k(\beta^m), \nabla_W F(W_k, Y_k)\rangle + \langle Y_k - Y_k(\beta^m), \nabla_Y F(W_k, Y_k)\rangle),$$

where $W_k(\alpha) \triangleq \max(W_k - \alpha \nabla_W F(W_k, Y_k), 0)$
and $Y_k(\alpha) \triangleq \max(Y_k - \alpha \nabla_Y F(W_k, Y_k), 0)$.
Let $W_{k+1} := W_k(\beta^{m_k})$; $Y_{k+1} := Y_k(\beta^{m_k})$.
end for

3. **LBNCG**: Log-barrier Newton-conjugate-gradient [22]. This method is equipped with worst case complexity guarantees (see Table 1) but its practical performance has not been studied to date. We implement it as Algorithm 1 in [22] with parameter choices $\epsilon_g = 10^{-4}$, $\theta = 1/2$, $\xi_r = 1/2$, $\bar{\xi} = 1/2$, $\beta = 1/2$, $\eta = 1/2$. We deal with the CG accuracy tolerance $\hat{\xi}_r$ and c_μ similarly as in our implementation of **PNCG**, setting them initially to .1 and decreasing them when we find that the modified CG is not yielding descent directions. Similar as in PNCG, we turn off Procedure 3 (MEO) in Algorithm 1 in [22] because we are only interested in locating an approximate first-order solution. Termination criterion is $\nabla F(W_k, Y_k) > -10^{-4}$ and $|\min\{[W_k^T, Y_k], 1\} \odot [\nabla_W F(W_k, Y_k)^T, \nabla_Y F(W_k, Y_k)]| \leq 10^{-4}$, where $>, \leq, \min\{\}, |\cdot|$ hold elementwisely and $\odot$ denotes elementwise multiplication.

An outer-loop iteration limit of 5000 and a running time of 100s are set for **PNCG** and **pgrad**. An outer-loop iteration limit of 10000 and a time limit of 60s are applied to **LBNCG**.

Experiment settings and metrics. To create Table 2, we generate three different scenarios ($(m,n,r) \in \{(150,100,15), (300,200,15), (600,400,15)\}$). The

Table 2 Comparison of three solvers with complexity guarantees on NMF. Three scenarios are considered with different dimensions m and n. In each scenario, 5 trials are run from different initial points (each of the four algorithms starts from the same initial point on each trial) and average results are reported. Elapsed time of each algorithm is reported. F^* is the objective function value of the output. **residual** is defined in (39) and **projnorm** represents $\|\nabla^P F(W,Y)\|_F$. **PNCG** and **pgrad** have similar performance that clearly dominates **LBNCG**, which always fails to converge in the allotted time / iteration limit.

Algorithm	outer-loop iteration	time(s)	F^*	residual	projnorm
		$m = 150$, $n = 100$, $r = 15$			
PNCG	1030.4	1.3	15.8	2.7e-05	6.3e-05
pgrad	1275.2	1.4	15.8	9.4e-05	9.5e-05
LBNCG	9774.8	57.7	4574.0	2.6e+04	4.4e+04
		$m = 300$, $n = 200$, $r = 15$			
PNCG	639.4	2.3	68.9	2.8e-05	1.4e-04
pgrad	708.8	2.2	68.9	9.3e-05	9.5e-05
LBNCG	4529.4	60.0	23702.8	7.5e+04	1.4e+05
		$m = 600$, $n = 400$, $r = 15$			
PNCG	579.2	8.1	285.5	3.0e-05	1.1e-04
pgrad	619.4	8.8	285.5	9.3e-05	9.7e-05
LBNCG	1364.8	60.0	146213.1	2.1e+05	4.3e+05

elements of the initial matrices W_0 and Y_0 are chosen from the half standard normal distribution, then normalized so that the average elementwise magnitude of either W_0 or Y_0 is 1. Given $\bar{x} \geq 0$, the residual of (1),(2) is defined following Definition 1:

$$\text{residual} = \max\left\{\|\bar{S}\nabla f(\bar{x})\|, -\min_{i\in J^+}\{\nabla_i f(\bar{x})\}\right\}, \tag{39}$$

where $J^+ \triangleq \{i \in \mathcal{I} \mid 0 \leq \bar{x}^i \leq \sqrt{\epsilon_r}\}$, $J^- \triangleq \{1,\ldots,n\} \setminus J^+ = \mathcal{I}^c \cup \{i \in \mathcal{I} \mid \bar{x}^i > \sqrt{\epsilon_r}\}$, and $\bar{S} = \text{diag}(\bar{s})$ is a diagonal matrix with $\bar{s}^i = 1$ when $i \in J^-$ and $\bar{s}^i = \bar{x}^i$ when $i \in J^+$. In this experiment we let $\epsilon_r = 10^{-6}$.

Results. Table 2 indicates that **PNCG** and **pgrad** are close in performance, with **PNCG** attaining slightly better residual measures. **PNCG** requires fewer outer-loop iterations because the Newton-CG steps taken on some iterations yield more progress than a first-order step. **LBNCG** is not competitive, perhaps not surprisingly since this method was designed with good **worst-case** complexity in mind, rather than for any practical considerations.

5.2 Comparison with specialized NMF schemes

We now compare **PNCG** with efficient alternating-direction schemes that are specialized for NMF. The following methods are compared.

1. **PNCG**(Algorithm 1): We use the same settings as in in Section 5.1, except that the MEO Procedure (Procedure (4)) is turned on and implemented using CG (see [23, Theorem 1]) with $\delta = .01$. This procedure enables PNCG to escape from a saddle point, as is shown in Figure 1(d).
2. **alspgrad**: Alternating nonnegative least squares using projected gradient, described in [19]. Parameter settings are as described in [19], except that the algorithm is stopped when $\|\nabla^P F(W,Y)\|_F \leq 10^{-4}$ (instead of $\|\nabla^P F(W,Y)\|_F \leq 10^{-4} \times \|\nabla F(W_0,Y_0)\|_F$) and the initial tolerance for the subproblem is set as 10^{-3} (instead of $10^{-3} \times \|\nabla F(W_0,Y_0)\|_F$).
3. **pnm**: Alternating nonnegative least squares using two-metric gradient projection, described in [14]. Parameter settings from [14] are used, except that the algorithm is stopped when $\|\nabla^P F(W,Y)\|_F \leq 10^{-4}$ and the initial tolerance for the subproblem is set as 10^{-3}.

An outer-loop iteration limit of 5000 and a running time limit of 100s are set for **PNCG**, while limits of 1000 and 100s are applied to the other two algorithms.

Settings. Synthetic datasets are created as above, with $m = 300$ and $n = 200$, and $r = 5, 10, 15$. To create plots (a)-(c) in Figure 1, we use the same initial matrices W_0, Y_0 for all algorithms, generated i.i.d. elementwise from a half standard normal distribution, then normalized such that the average magnitude of either W_0 or Y_0 is 1. For plot (d), we start both algorithms near a saddle point of (37), constructed according to the following observation. If $w \in \mathbb{R}^{m\times 1}$ and $y \in \mathbb{R}^{1\times n}$ constitute a first-order optimal point of (37) (that is, $\nabla^P F(w,y) = 0$) when $r = 1$, then

$$W \triangleq \frac{1}{5} w \times \underbrace{(1,\ldots,1)}_{10}, \quad Y \triangleq \frac{1}{2} \underbrace{(1;\ldots;1)}_{10} \times y \tag{40}$$

constitute a first-order optimal point of (37) when $r = 10$. In the experiment, we first use **alspgrad** to solve (37) with $r = 1$ and obtain the approximate solution w and y. We then set W_0 and Y_0 as in (40), and run **alspgrad** with $r = 10$ from this starting point, to see if it is able to escape from the saddle point. The other approaches are run from the same choice of W_0 and Y_0.

Results. Plots of function value against running time are generated in Figure 1. In plots (a)-(c), we see that **PNCG** is almost comparable with **alspgrad** in minimizing the objective function and slightly worse than **pnm**. When m and n are larger than the values used here, **PNCG** becomes less competitive. In fact, the cost of the Hessian-vector product or gradient evaluation or checking step acceptance criterion in **PNCG** is $O(mnr)$; the cost of the gradient evaluation or step acceptance criterion validation in the subproblem of **alspgrad** is either $\mathcal{O}(mr^2)$ or $\mathcal{O}(nr^2)$; the cost of gradient evaluation or partial Hessian evaluation or step acceptance criterion validation in **pnm** is either $\mathcal{O}(mr^2)$ or $\mathcal{O}(nr^2)$, while the step direction calculation in **pnm** costs either $\mathcal{O}(\bar{m}\bar{s}^3)$ or $\mathcal{O}(\bar{n}\bar{s}^3)$

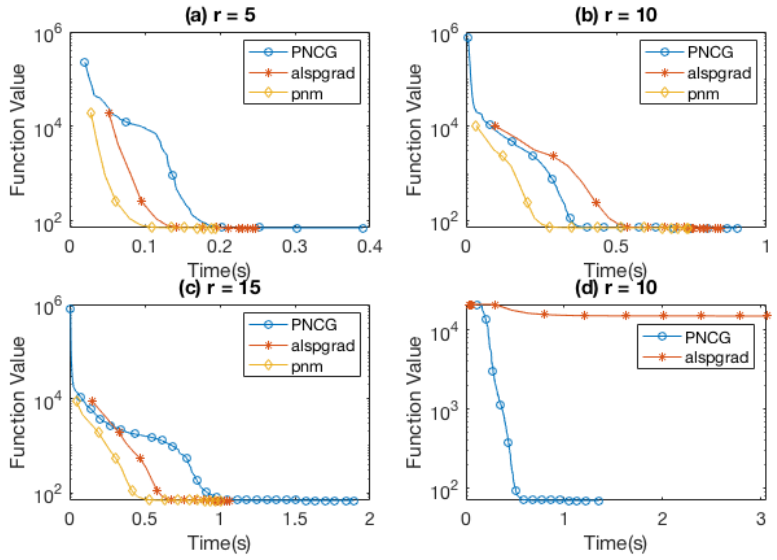

Fig. 1 Comparison between **PNCG** and two solvers that are specialized to NMF, showing the objective function value plotted against elapsed time. We set $m = 300$ and $n = 200$ use three different values of r in plots (a)-(c), with initial points generated randomly. In plot (d), the algorithms are started near a saddle point. Note that all algorithms start from the same initial point. The plots appear not so because we draw the curves after the algorithms make progress.

where $\bar{m} \leq m$, $\bar{n} \leq n$, $\bar{s} \leq r$. Therefore, when $m \gg r$, $n \gg r$, the higher costs of these basic operations compromise the performance of PNCG.

In plot (d), where the algorithms are initialized near the saddle point, **pnm** cannot be applied since the Hessian for the subproblem is singular at the initial point. The first-order method **alspgrad** is able to escape from the vicinity of the current saddle point and reduce the objective a little, but it appears to get stuck at another suboptimal point. Meanwhile, **PNCG** appears to exit the saddle point, due to a call to the MEO procedure, Algorithm 4. We include plot (d) to verify the theory for **PNCG** in the worst-case scenario of starting at a saddle point. Random starts like those used in the other plots are likely to yield convergence of the specialized methods to local minima.

6 Conclusion

In this article, we relate and compare different definitions of approximate second-order optimal point in literature and define our own for optimization with bounds. We proposed a projected Newton-CG method. It has good complexity guarantees and is designed with practicality in mind, and is related to the two-metric projection algorithms proposed in the 1980s. The projected Newton-CG terminates within $\mathcal{O}(\epsilon^{-3/2})$ iterations or $\tilde{\mathcal{O}}(\epsilon^{-7/4})$ number

of Hessian-vector product/gradient evaluation operations and finds a point that is approximately second-order optimal to tolerance ϵ, with high probability. Numerical experiments on nonnegative matrix factorization illustrate practicality of the methods.

In future work, we will consider extensions of the algorithms to solve optimization problems with more complex structures such as linear inequality constraints and ℓ_1-norm terms. We will also investigate complexity guarantees of the two-metric projection algorithm proposed by Bertsekas.

References

1. Bertsekas, D.P.: Projected Newton methods for optimization problems with simple constraints. SIAM Journal on Control and Optimization **20**(2), 221–246 (1982). DOI 10.1137/0320018
2. Bertsekas, D.P.: Constrained optimization and Lagrange multiplier methods. Academic Press (2014)
3. Bertsekas, D.P.: Nonlinear programming, third edn. Athena Scientific, Belmont, MA 02478 (2016)
4. Bian, W., Chen, X., Ye, Y.: Complexity analysis of interior point algorithms for non-lipschitz and nonconvex minimization. Mathematical Programming **149**(1), 301–327 (2015). DOI 10.1007/s10107-014-0753-5
5. Birgin, E.G., Martínez, J.M.: Complexity and performance of an augmented lagrangian algorithm. Optimization Methods and Software **35**(5), 885–920 (2020)
6. Birgin, E.G., Martínez, J.M.: On regularization and active-set methods with complexity for constrained optimization. SIAM Journal on Optimization **28**(2), 1367–1395 (2018)
7. Cartis, C., Gould, N.I., Toint, P.L.: 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)
8. Cartis, C., Gould, N.I., Toint, P.L.: 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)
9. Cartis, C., Gould, N.I., Toint, P.L.: Second-order optimality and beyond: Characterization and evaluation complexity in convexly constrained nonlinear optimization. Foundations of Computational Mathematics **18**(5), 1073–1107 (2018)
10. Curtis, F.E., Robinson, D.P., Samadi, M.: Complexity analysis of a trust funnel algorithm for equality constrained optimization. SIAM Journal on Optimization **28**(2), 1533–1563 (2018)
11. Dvurechensky, P., Staudigl, M.: Hessian barrier algorithms for non-convex conic optimization. arXiv preprint arXiv:2111.00100 (2021)
12. Ghadimi, S., Lan, G., Zhang, H.: Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. Mathematical Programming **155**(1), 267–305 (2016)
13. Gillis, N.: The why and how of nonnegative matrix factorization. In: J.A.K. Suykens, M. Signoretto, A. Argyriou (eds.) Regularization, Optimization, Kernels, and Support Vector Machines, Machine Learning and Pattern Recognition, pp. 257–291. Chapman & Hall/CRC (2014)
14. Gong, P., Zhang, C.: Efficient nonnegative matrix factorization via projected newton method. Pattern Recognition **45**(9), 3557–3565 (2012)
15. Grapiglia, G.N., Yuan, Y.x.: On the complexity of an augmented lagrangian method for nonconvex optimization. IMA Journal of Numerical Analysis **41**(2), 1546–1568 (2021)
16. Griewank, A., Walther, A.: Evaluating derivatives: principles and techniques of algorithmic differentiation. SIAM (2008)
17. Haeser, G., Liu, H., Ye, Y.: Optimality condition and complexity analysis for linearly-constrained optimization without differentiability on the boundary. Mathematical Programming (2018). DOI 10.1007/s10107-018-1290-4

18. Kim, J., Park, H.: Toward faster nonnegative matrix factorization: A new algorithm and comparisons. In: 2008 Eighth IEEE International Conference on Data Mining, pp. 353–362. IEEE (2008)
19. Lin, C.J.: Projected gradient methods for nonnegative matrix factorization. Neural computation **19**(10), 2756–2779 (2007)
20. Lin, Q., Ma, R., Xu, Y.: Complexity of an inexact proximal-point penalty method for constrained smooth non-convex optimization. Computational Optimization and Applications pp. 1–50 (2022)
21. Moré, J.J., Toraldo, G.: On the solution of large quadratic programming problems with bound constraints. SIAM Journal on Optimization **1**(1), 93–113 (1991). DOI 10.1137/0801008. URL https://doi.org/10.1137/0801008
22. O'Neill, M., Wright, S.J.: A log-barrier Newton-CG method for bound constrained optimization with complexity guarantees. IMA Journal of Numerical Analysis (2020). DOI 10.1093/imanum/drz074
23. Royer, C.W., O'Neill, M., Wright, S.J.: A Newton-CG algorithm with complexity guarantees for smooth unconstrained optimization. Mathematical Programming (2019). DOI 10.1007/s10107-019-01362-7
24. Royer, C.W., Wright, S.J.: Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization. SIAM Journal on Optimization **28**(2), 1448–1477 (2018). DOI 10.1137/17M1134329
25. Sahin, M.F., Alacaoglu, A., Latorre, F., Cevher, V., et al.: An inexact augmented lagrangian framework for nonconvex optimization with nonlinear constraints. Advances in Neural Information Processing Systems **32** (2019)
26. Xie, Y., Wright, S.J.: Complexity of proximal augmented lagrangian for nonconvex optimization with nonlinear equality constraints. Journal of Scientific Computing **86**(3), 1–30 (2021)

A Comparing approximate second-order optimality conditions

We show that the relation $\gtrsim$ is transitive.

Lemma 6 *If* **DefA** $\gtrsim$ **DefB** *and* **DefB** $\gtrsim$ **DefC**, *then* **DefA** $\gtrsim$ **DefC**.

Proof Since **DefA** $\gtrsim$ **DefB**, there exists $\epsilon_A \in (0,1]$ and $c_A > 0$ such that for any $x \in \mathcal{X}$ and $\epsilon \in (0, \epsilon_A]$, if x is an (ϵ, p)-2o by **DefA**, then x is also a $(c_A\epsilon, p)$-2o by **DefB**.

Likewise, since **DefB** $\gtrsim$ **DefC**, there exists $\epsilon_B \in (0,1]$ and $c_B > 0$ such that for any $x \in \mathcal{X}$ and $\epsilon \in (0, \epsilon_B]$, if x is an (ϵ, p)-2o by **DefB**, then x is also a $(c_B\epsilon, p)$-2o by **DefC**.

Let $\bar{\epsilon} = \min\{\epsilon_A, \epsilon_B/c_A\}$. Take arbitrary $x \in \mathcal{X}$ and $\epsilon \in (0, \bar{\epsilon}]$. Suppose that x is an (ϵ, p)-2o by **DefA**. Since **DefA** $\gtrsim$ **DefB** and $\epsilon \leq \epsilon_A$, x is a $(c_A\epsilon, p)$-2o by **DefB**; Since **DefB** $\gtrsim$ **DefC** and $c_A\epsilon \leq \epsilon_B$, x is also a $(c_Bc_A\epsilon, p)$-2o by **DefC**. Since the choice of $x \in \mathcal{X}$ and $\epsilon \in (0, \bar{\epsilon}]$ is arbitrary, we have that **DefA** $\gtrsim$ **DefC**. □

We now give the proof of Theorem 2.

Proof Let $U_{\mathcal{X}} \triangleq \max_{x\in\mathcal{X}} \|x\|_\infty$.

(1) Suppose that x is an (ϵ, p)-2o by **Def2** and $\epsilon \leq 1$. We show (1) through the following steps.
  (1a) Fix any index i. Choose d such that $d^i = \Delta$ and $d^j = 0, \forall j \neq i$, then (12) $\implies$ $\nabla f(x)^T d = \nabla_i f(x)\Delta \geq -\Delta\epsilon \implies \nabla_i f(x) \geq -\epsilon$. This indicates that $\nabla f(x) \geq -\epsilon\mathbf{1}$.
  (1b) Fix any index i. Let $d^i = -\text{sign}(\nabla_i f(x)) \min\{\Delta, x^i\}$ and $d^j = 0$, $\forall j \neq i$. Then (12) $\implies$ $\nabla f(x)^T d = -|\nabla_i f(x)| \min\{\Delta, x^i\} \geq -\Delta\epsilon \implies |\nabla_i f(x)| \leq \frac{\Delta\epsilon}{\min\{\Delta, x^i\}}$ $\implies |\nabla_i f(x) x^i| \leq \frac{\Delta x^i \epsilon}{\min\{\Delta, x^i\}} = \max\{x^i, \Delta\}\epsilon \leq \max\{U_{\mathcal{X}}, \Delta\}\epsilon$. This indicates that $\|X\nabla f(x)\|_\infty \leq \max\{U_{\mathcal{X}}, \Delta\}\epsilon \leq \max\{U_{\mathcal{X}}, \Delta_{\max}\}\epsilon$.

(1c) If $x = 0$, then second row of (13) holds trivially. Suppose that $x \neq 0$. Let $d \triangleq c_d X v$, where $c_d \triangleq \min\left\{\frac{\Delta}{\|x\|_\infty}, 1\right\}$, v is an arbitrary vector with $\|v\|_2 = 1$. Therefore, we have that $x + d \geq 0, x - d \geq 0, \|d\| \leq \Delta$. Therefore,

$$-\Delta^2\epsilon^p \overset{(12)}{\leq} \nabla f(x)^T d + \frac{1}{2} d^T \nabla^2 f(x) d$$

and also,

$$-\Delta^2\epsilon^p \overset{(12)}{\leq} -\nabla f(x)^T d + \frac{1}{2} d^T \nabla^2 f(x) d.$$

Therefore,

$$-\Delta^2\epsilon^p \leq \frac{1}{2} d^T \nabla^2 f(x) d = \frac{c_d^2}{2} v^T X \nabla^2 f(x) X v$$
$$\implies v^T X \nabla^2 f(x) X v \geq -\frac{2\Delta^2}{c_d^2}\epsilon^p \geq -2\max\{\|x\|_\infty^2, \Delta^2\}\epsilon^p \geq -2\max\{U_{\mathcal{X}}^2, \Delta_{\max}^2\}\epsilon^p.$$

Denote $c_\Delta \triangleq \left(2\max\{U_{\mathcal{X}}^2, \Delta_{\max}^2\}\right)^{1/p}$. Therefore, by (1a)-(1c), x is an $(c\epsilon, p)$-2o by **Def3**, where $c \triangleq \max\{1, U_{\mathcal{X}}, \Delta_{\max}, c_\Delta\}$.

(2) Given $x \geq 0$, let $T \triangleq \mathrm{diag}(t)$ be a diagonal matrix of $n \times n$ such that $t^i = 1$ if $x^i \leq 1$ and $t^i = 1/x^i$ if $x^i > 1$. Then we have that $\bar{X} = XT = TX$.
Def3 $\gtrsim$ **Def4**. Suppose that x is an (ϵ, p)-2o by **Def3**. Note

$$\|\bar{X}\nabla f(x)\|_\infty \leq \|X\nabla f(x)\|_\infty \leq \epsilon.$$
$$d^T\bar{X}\nabla^2 f(x)\bar{X}d = (Td)^T X\nabla^2 f(x) XTd \geq -\epsilon^p\|Td\|^2 \geq -\epsilon^p\|d\|^2, \forall d \in \mathbb{R}^n.$$

Therefore, x is also an (ϵ, p)-2o by **Def4**.
Def4 $\gtrsim$ **Def3**. x is an (ϵ, p)-2o by **Def4**. Then

$$\|X\nabla f(x)\|_\infty = \|T^{-1}\bar{X}\nabla f(x)\|_\infty \leq \|x\|_\infty\|\bar{X}\nabla f(x)\|_\infty \leq U_{\mathcal{X}}\epsilon.$$
$$d^T X\nabla^2 f(x) X d = (T^{-1}d)^T\bar{X}\nabla^2 f(x)\bar{X}(T^{-1}d) \geq -\epsilon^p\|T^{-1}d\|^2$$
$$\geq -\epsilon^p\|x\|_\infty^2\|d\|^2 \geq -(U_{\mathcal{X}}^{2/p}\epsilon)^p\|d\|^2.$$

Then x is also an $(\max\{U_{\mathcal{X}}, U_{\mathcal{X}}^{2/p}\}\epsilon, p)$-2o by **Def3**.

(3) Obviously we have that **Def3** $\gtrsim$ **Def5**. By (2) and the property of $\gtrsim$ and $\approx$, **Def4** $\gtrsim$ **Def5**.

(4) Suppose that x is an (ϵ, p)-2o by **Def1**. Let J^+, J^-, and S be associated with x as in **Def1**. Let $T = \mathrm{diag}(t)$ be a diagonal matrix of dimension $n \times n$ with $t^i = 1$ for $i \in J^+$ and $t^i = x^i$, for $i \in J^-$. Then $X = TS$ and

$$\|X\nabla f(x)\|_\infty \leq \|X\nabla f(x)\| = \|TS\nabla f(x)\| \leq \|T\|\|S\nabla f(x)\|$$
$$\leq \max\{\|x\|_\infty, 1\}\|S\nabla f(x)\| \leq 2\max\{U_{\mathcal{X}}, 1\}\epsilon \leq \max\{U_{\mathcal{X}}^{2/p}, 2U_{\mathcal{X}}, 2\}\epsilon.$$

Also, for any $d \in \mathbb{R}^n$, we have

$$d^T X\nabla^2 f(x) X d = d^T TS\nabla^2 f(x) STd \geq -\epsilon^p\|Td\|^2$$
$$\geq -\epsilon^p\max\{\|x\|_\infty^2, 1\}\|d\|^2 = -(\epsilon\max\{\|x\|_\infty^{2/p}, 1\})^p\|d\|^2$$
$$\geq -(\epsilon\max\{U_{\mathcal{X}}^{2/p}, 1\})^p\|d\|^2 \geq -(\epsilon\max\{U_{\mathcal{X}}^{2/p}, 2U_{\mathcal{X}}, 2\})^p\|d\|^2.$$

If we let $c = \max\{U_{\mathcal{X}}^{2/p}, 2U_{\mathcal{X}}, 2\}$, then x is an $(c\epsilon, p)$-2o by **Def5**.

□

The following example illustrates why **Def2**,**Def3**,**Def4** are not essentially stronger than **Def1**.

Example 1 Consider problem (1),(2),(11) in 1-dimension. Let $f(x) = \frac{1}{4}x^4$ and $p \in (0, 1]$. Given any $c > 0$, there exists $\bar{\epsilon} \in (0, 1)$ such that for any $\epsilon \in (0, \bar{\epsilon})$, we can find an $x \geq 0$ that is (ϵ, p)-2o by **Def2**, **Def3**, **Def4**, but not a $(c\epsilon, p)$-2o by **Def1**. In particular, choose $\bar{\epsilon}$ such that for any $\epsilon \in (0, \bar{\epsilon})$,

$$\epsilon^{7/24} \leq \Delta_{\max},\ \epsilon^{1/6} \leq \Delta_{\max},\ \epsilon^{1/6} \leq 6\Delta_{\max}^2,\ \epsilon^{5/24} < c^{-1/2},\ \epsilon^{1/8} < \frac{1}{2c}. \tag{41}$$

Let $x = \epsilon^{7/24}$. Then $f'(x) = x^3 = \epsilon^{7/8}, xf'(x) = x^4 = \epsilon^{7/6}, f''(x) = 3x^2 \geq 0, \forall x \geq 0$. Apparently, x is an (ϵ, p)-2o by **Def3**, **Def4**. Note that by (41),

$$\begin{aligned}
&\min_{x+d\geq 0, |d|\leq \Delta_{\max}} f'(x)d = -x^3 \cdot x = -\epsilon^{7/6} \geq -\Delta_{\max}\epsilon.\\
&\min_{x+d\geq 0, |d|\leq \Delta_{\max}} f'(x)d + \frac{1}{2}f''(x)d^2 = \min_{x+d\geq 0, |d|\leq \Delta_{\max}} x^3 d + \frac{3x^2}{2}\cdot d^2\\
&\overset{(d^*=-x/3)}{=} -x^4/6 = -\epsilon^{7/6}/6 \geq -\Delta_{\max}^2\epsilon^p.
\end{aligned}$$

Therefore, x is also an (ϵ, p)-2o by **Def2**(let $\Delta = \Delta_{\max}$). However, note that by (41),

$$x > \sqrt{c\epsilon} \implies J^- = \{1\};\ f'(x) = \epsilon^{7/8} > 2c\epsilon,$$

so x is not an $(c\epsilon, p)$-2o by **Def1**.

B Capped conjugate gradient algorithm

The version of the conjugate gradient method shown in Algorithm 3 was described in [23, Algorithm 1] to solve a system of the form $\bar{H}y = -g$, where $\bar{H} = H + 2\epsilon I$ is a damped version of the symmetric matrix H, which in our case is a principal submatrix of the Hessian $\nabla^2 f(x_k)$. Note that the following results hold regarding Algorithm 3, as is shown in [23, Lemma 3] and [23, Lemma 1].

Lemma 7 *Consider the inputs H, g, ϵ and outputs d_type and d of Algorithm 3, if*
1. d_type=SOL, then

$$d^T(H + 2\epsilon I)d \geq \epsilon\|d\|^2, \quad \|d\| \leq 1.1\epsilon^{-1}\|g\|, \quad \|r\| \leq \frac{1}{2}\epsilon\zeta\|d\|,$$

where $r \triangleq (H + 2\epsilon I)d + g$.
2. d_type = NC and $\bar{d} \triangleq -\operatorname{sgn}(d^T g)\frac{d^T H d}{\|d\|^2}\frac{d}{\|d\|}$, then $\bar{d}^T g \leq 0$ and

$$\frac{\bar{d}^T H \bar{d}}{\|\bar{d}\|^2} = -\|\bar{d}\| \leq -\epsilon.$$

Lemma 8 *Number of matrix-vector multiplication of Algorithm 3 is bounded by*

$$2\min\{n, \mathbb{J}(M, \epsilon, \zeta)\} + 1,$$

where

$$\mathbb{J}(M, \epsilon, \zeta) \leq \min\left\{n, \left\lceil\left(\sqrt{\kappa} + \frac{1}{2}\right)\log\left(\frac{144(\sqrt{\kappa}+1)^2\kappa^6}{\zeta^2}\right)\right\rceil\right\}.$$

Algorithm 3 Capped Conjugate Gradient

Inputs: Symmetric matrix $H \in \mathbb{R}^{n\times n}$; vector $g \neq 0$; damping parameter $\epsilon \in (0,1)$; desired relative accuracy $\zeta \in (0,1)$;
Optional input: scalar M (set to 0 if not provided);
Outputs: d_type, d;
Secondary outputs: final values of M, κ, $\hat{\zeta}$, τ, and T;
Set

$$\bar{H} := H + 2\epsilon I, \quad \kappa := \frac{M+2\epsilon}{\epsilon}, \quad \hat{\zeta} := \frac{\zeta}{3\kappa}, \quad \tau := \frac{\sqrt{\kappa}}{\sqrt{\kappa}+1}, \quad T := \frac{4\kappa^4}{(1-\sqrt{\tau})^2};$$

Set $y_0 \leftarrow 0$, $r_0 \leftarrow g$, $p_0 \leftarrow -g$, $j \leftarrow 0$;
if $p_0^\top \bar{H} p_0 < \epsilon \|p_0\|^2$ **then**
  Set $d = p_0$ and terminate with d_type=NC;
else if $\|Hp_0\| > M\|p_0\|$ **then**
  Set $M \leftarrow \|Hp_0\|/\|p_0\|$ and update $\kappa, \hat{\zeta}, \tau, T$ accordingly;
end if
while TRUE **do**
  $\alpha_j \leftarrow r_j^\top r_j / p_j^\top \bar{H} p_j$; {Begin Standard CG Operations}
  $y_{j+1} \leftarrow y_j + \alpha_j p_j$;
  $r_{j+1} \leftarrow r_j + \alpha_j \bar{H} p_j$;
  $\beta_{j+1} \leftarrow (r_{j+1}^\top r_{j+1})/(r_j^\top r_j)$;
  $p_{j+1} \leftarrow -r_{j+1} + \beta_{j+1} p_j$; {End Standard CG Operations}
  $j \leftarrow j+1$;
  if $M < \max(\|Hp_j\|/\|p_j\|, \|Hy_j\|/\|y_j\|, \|Hr_j\|/\|r_j\|)$ **then**
    Set $M \leftarrow \max(\|Hp_j\|/\|p_j\|, \|Hy_j\|/\|y_j\|, \|Hr_j\|/\|r_j\|)$ and update $\kappa, \hat{\zeta}, \tau, T$ accordingly;
  end if
  if $y_j^\top \bar{H} y_j < \epsilon \|y_j\|^2$ **then**
    Set $d \leftarrow y_j$ and terminate with d_type=NC;
  else if $\|r_j\| \le \hat{\zeta}\|r_0\|$ **then**
    Set $d \leftarrow y_j$ and terminate with d_type=SOL;
  else if $p_j^\top \bar{H} p_j < \epsilon \|p_j\|^2$ **then**
    Set $d \leftarrow p_j$ and terminate with d_type=NC;
  else if $\|r_j\| > \sqrt{T}\tau^{j/2}\|r_0\|$ **then**
    Compute α_j, y_{j+1} as in the main loop above;
    Find $i \in \{0, \ldots, j-1\}$ such that

$$\frac{(y_{j+1}-y_i)^\top \bar{H}(y_{j+1}-y_i)}{\|y_{j+1}-y_i\|^2} < \epsilon; \tag{42}$$

    Set $d \leftarrow y_{j+1} - y_i$ and terminate with d_type=NC;
  end if
end while

C Minimum eigenvalue oracle (MEO) procedure

The procedure shown as Procedure 4 is used to identify a direction of significant negative curvature, smaller than a threshold $-\epsilon/2$ for a given $\epsilon > 0$, or else return a certificate that all eigenvalues of H are greater than $-\epsilon$. In the latter case, the certificate may be wrong, with probability up to a supplied tolerance δ. This procedure is defined in [23, Procedure 2], where a discussion of various possible implementations is given. The most interesting implementation for our purposes is a randomized Lanczos procedure, which performs a single matrix-vector product involving H at each of its iterations, and which finds the minimum

eigenvalue of the projection of H onto a Krylov subspace seeded by an initial random vector at each iteration.

Procedure 4 Minimum Eigenvalue Oracle (MEO)

Inputs: Symmetric matrix $H \in \mathbb{R}^{n\times n}$, tolerance $\epsilon > 0$, error probability $\delta \in [0, 1)$;
Optional input: Upper bound on Hessian norm M;
Outputs: An estimate λ of $\lambda_{\min}(H)$ such that $\lambda \le -\epsilon/2$, and vector v with $\|v\| = 1$ such that $v^\top Hv = \lambda$ **OR** a certificate that $\lambda_{\min}(H) \ge -\epsilon$. The probability that the certificate is issued but $\lambda_{\min}(H) < -\epsilon$ is at most δ.

Based on the discussion in [23, Section 3.2] and [23, Assumption 3], we have the following result about bound on Hessian-vector products when a randomized Lanczos procedure (or a randomized CG) is used to implement Procedure 4.

Lemma 9 *We use a randomized Lanczos method with a starting vector uniformly generated on a unit sphere to implement Procedure 4. Then given a failure probability $0 < \delta \ll 1$, Procedure 4 either certifies that $H \succeq -\epsilon I$ (with failure probability $1 - \delta$) or finds a direction along which curvature of H is smaller than $-\epsilon/2$ in at most $N^{\text{meo}} \triangleq \min\{n, 1 + \lceil \mathcal{C}^{\text{meo}}\epsilon^{-1/2}\rceil\}$ Hessian-vector products, where $\mathcal{C}^{\text{meo}} = \log(2.75n/\delta^2)\sqrt{\|H\|\max\{1, \epsilon^2\}}/2$.*

D Two-sided bounds

In this section we consider the two-sided bound-constrained optimization:

$$\min f(x) \quad \text{s.t. } x \in \Omega \triangleq \{x \in \mathbb{R}^n \mid 0 \le x^i \le u^i,\ i \in \mathcal{I}\} \tag{43}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable and is bounded below by f_{low} on the feasible region Ω, and $\mathcal{I} \subset \{1, 2, \ldots, n\}$. We assume without loss of generality that $u^i > 0$ for all $i \in \mathcal{I}$. We allow $u_i = \infty$, that is, not all components x_i for $i \in \mathcal{I}$ have upper bounds.

Extending Definition 1, we define approximate optimality for (43) as follows.

Definition 7 ((ϵ, p)-2o approximate optimality of (43)) We say that x is an (ϵ, p)-2o approximately optimal point for (43) if and only if

$$0 \le x^i \le u^i,\ i \in \mathcal{I},\ \|S\nabla f(x)\| \le 2\epsilon, \text{ and } \begin{cases} \nabla_i f(x) \ge -\epsilon^{3/4}, & i \in \mathcal{I}, x_k^i \le \sqrt{\epsilon}, \\ \nabla_i f(x) \le \epsilon^{3/4}, & i \in \mathcal{I}, x_k^i \ge u^i - \sqrt{\epsilon}, \end{cases}$$
$$S\nabla^2 f(x)S \succeq -\epsilon^p I,$$

where we define $J^+ \triangleq \{i \in \mathcal{I} \mid 0 \le x^i \le \sqrt{\epsilon} \text{ or } u^i - \sqrt{\epsilon} \le x^i \le u^i\}$, $J^- \triangleq \{1, \ldots, n\} \setminus J^+$, $S = \text{diag}(s)$, where $s^i = \min\{x^i, u^i - x^i\}$ if $i \in J^+$, and $s^i = 1$ if $i \in J^-$. Again, this definition reduces to Definition 1 when $u^i = +\infty$ for all $i \in \mathcal{I}$, and can be motivated by exact (weak) second-order optimal conditions of (43). The extension of projected Newton-CG (Algorithm 1) to the general bound-constrained optimization (43) is relatively straightforward. We redefine the projection operator P, index sets J_k^+ and J_k^-, and $S_k = \text{diag}(s_k)$ as follows:

$$[P(x)]^i \triangleq \begin{cases} \text{mid}(0, x^i, u^i) & i \in \mathcal{I}, \\ x^i & \text{otherwise}, \end{cases}$$
$$J_k^+ \triangleq \{i \in \mathcal{I} \mid 0 \le x_k^i \le \epsilon_k \text{ or } u^i - \epsilon_k \le x_k^i \le u^i\},$$
$$J_k^- \triangleq \{1, \ldots, n\} \setminus J_k^+ = \{i \in \mathcal{I} \mid \epsilon_k < x_k^i < u^i - \epsilon_k\} \cup \mathcal{I}^c.$$

$$s_k^i = \begin{cases} \min\{x_k^i, u^i - x_k^i\}, & \text{if } i \in J_k^+, \\ 1, & \text{otherwise.} \end{cases}$$

The definitions of g_k^-, H_k^-, g_k^+ and S_k^+, are the same, modulo the redefined P, J_k^+, J_k^-, and S_k. For Algorithm 1, the only adjustment to be made is the conditions to trigger the gradient step, which become

$$g_k^i < -\epsilon_k^{3/2}, x_k^i \le \epsilon_k, i \in \mathcal{I} \text{ or } g_k^i > \epsilon_k^{3/2}, x_k^i \ge u^i - \epsilon_k, i \in \mathcal{I} \text{ or } \|S_k^+ g_k^+\| \ge \epsilon_k^2.$$

We make an additional assumption on ϵ_k that

$$2\epsilon_k \le u^i, \quad \forall k \ge 0,\ i \in \mathcal{I},$$

and assume that Assumption 1 and 2 hold when Ω includes two-sided bounds. It can then be verified that Lemmas 3, 4 and 5 still hold for the modified Algorithm 1. Lemma 2 also holds if the conditions to trigger the gradient step is tailored accordingly. Furthermore, if we let $\epsilon_k \equiv \epsilon_H = \sqrt{\epsilon}$ and $\epsilon_g = \epsilon$, then the Algorithm stops within the same number of iterations specified in Theorem 3 ($\mathcal{O}(\epsilon^{-3/2})$) and locates an x that is an $(\epsilon, 1/2)$-2o point of (43) with probability at least $(1-\delta)^{K_{\text{pncg}}}$, where $\delta \in [0, 1)$ is the probability of failure in Procedure 4. Moreover, the complexity of fundamental operations (gradient evaluations or Hessian-vector products) is also $\tilde{O}(\epsilon^{-7/4})$.