Newton-CG methods for nonconvex unconstrained optimization with Hölder continuous Hessian

Chuan He* Zhaosong Lu†

November 17, 2023

Abstract

In this paper we consider a nonconvex unconstrained optimization problem minimizing a twice differentiable objective function with Hölder continuous Hessian. Specifically, we first propose a Newton-conjugate gradient (Newton-CG) method for finding an approximate first-order stationary point (FOSP) of this problem, assuming the associated the Hölder parameters are explicitly known. Then we develop a parameter-free Newton-CG method without requiring any prior knowledge of these parameters. To the best of our knowledge, this method is the first parameter-free second-order method achieving the best-known iteration and operation complexity for finding an approximate FOSP of this problem. Furthermore, we propose a Newton-CG method for finding an approximate second-order stationary point (SOSP) of the considered problem with high probability and establish its iteration and operation complexity. Finally, we present preliminary numerical results to demonstrate the superior practical performance of our parameter-free Newton-CG method over a well-known regularized Newton method.

Keywords Nonconvex unconstrained optimization, Newton-conjugate gradient method, Hölder continuity, iteration complexity, operation complexity

Mathematics Subject Classification 49M15, 49M37, 58C15, 90C25, 90C30

1 Introduction

In this paper we consider the nonconvex unconstrained optimization problem

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1}$$

where $f:\mathbb{R}^n\to\mathbb{R}$ is twice continuously differentiable and $\nabla^2 f$ is Hölder continuous in an open neighborhood of a level set of f (see Assumption 1 for details). Our goal is to propose *easily implementable* second-order methods with complexity guarantees, particularly, Newton-conjugate gradient (Newton-CG) methods for finding approximate first- and second-order stationary points of problem (1).

In recent years, there have been significant advancements in second-order methods with complexity guarantees for problem (1) when $\nabla^2 f$ is *Lipschitz continuous*. Notably, cubic regularized Newton methods [1, 6, 10, 26], trust-region methods [12, 13, 23], second-order line-search method [29], inexact regularized Newton method [14], quadratic regularization method [4], and Newton-CG method [28] were developed for finding an $(\epsilon,\sqrt{\epsilon})$-second-order stationary point (SOSP) x of problem (1) satisfying

$$\|\nabla f(x)\|\le\epsilon,\qquad \lambda_{\min}(\nabla^2 f(x))\ge-\sqrt{\epsilon},$$

where $\epsilon\in(0,1)$ is a tolerance parameter and $\lambda_{\min}(\cdot)$ denotes the minimum eigenvalue of the associated matrix. Under suitable assumptions, it was shown that these second-order methods achieve an iteration complexity of

*Department of Computer Science and Engineering, University of Minnesota, USA (email: he000233@umn.edu).

†Department of Industrial and Systems Engineering, University of Minnesota, USA (email: zhaosong@umn.edu).

$\mathcal{O}(\epsilon^{-3/2})$ for finding an $(\epsilon, \sqrt{\epsilon})$-SOSP, which has been proved to be optimal in [9, 11]. In addition to iteration complexity, operation complexity of the methods in [1, 6, 12, 28, 29], measured by the number of their fundamental operations, was also studied. Under suitable assumptions, it was shown that these methods achieve an operation complexity of $\widetilde{\mathcal{O}}(\epsilon^{-7/4})$ for finding an $(\epsilon, \sqrt{\epsilon})$-SOSP of problem (1) with high probability.[1] Similar operation complexity bounds have also been achieved by gradient-based methods (e.g., see [2, 7, 8, 19, 22, 24, 30]).

Nonetheless, there has been very little study on second-order methods for problem (1) – a nonconvex unconstrained optimization problem with Hölder continuous Hessian. To the best of our knowledge, the regularized Newton methods proposed in [16, 31] are the only existing second-order methods for problem (1). Specifically, the cubic regularized Newton method in [16] tackles problem (1) by solving a sequence of cubic regularized Newton subproblems. It is a parameter-free second-order method and does not require any prior information on the modulus H_ν and exponent ν associated with the Hölder continuity (see Assumption 1). Under mild assumptions, it was shown in [16] that this method enjoys an iteration complexity of

$$\mathcal{O}\left(H_\nu^{\frac{1}{1+\nu}}\epsilon^{-\frac{2+\nu}{1+\nu}}\right) \tag{2}$$

for finding an approximate first-order stationary point (FOSP) x of problem (1) satisfying $\|\nabla f(x)\| \leq \epsilon$. This iteration complexity matches the lower iteration complexity bound established in [11]. Yet, its operation complexity remains unknown. Moreover, this method requires solving many cubic regularized Newton subproblems *exactly* per iteration, which is highly expensive to implement in general. In [31], two nice adaptive regularized Newton methods were proposed for finding an approximate SOSP of the problem minimizing a nonconvex function with Hölder continuous Hessian on a Riemannian manifold, which includes problem (1) as a special case. More specifically, when applied to problem (1), one method in [31] solves a sequence of $(2+\nu)$th-order regularized Newton subproblems, while another method in [31] solves a sequence of standard trust-region subproblems. Iteration and operation complexity results of these methods were established in [31]. However, these methods are *not fully parameter-free* since prior knowledge of the Hölder exponent is required in order to achieve the best-known complexity.

As discussed above, the existing second-order methods [16, 31] for problem (1) require solving a sequence of sophisticated trust-region or regularized Newton subproblems. In this paper, we propose *easily implementable* second-order methods, particularly Newton-CG methods for (1), by applying the capped CG method [28, Algorithm 1] to solve a sequence of systems of linear equations with coefficient matrix resulting from a proper perturbation on the Hessian of f. Specifically, we first propose a Newton-CG method (Algorithm 1) to find an approximate FOSP of (1), assuming the parameters associated with the Hölder continuity of $\nabla^2 f$ are explicitly known. Then we develop a *parameter-free* Newton-CG method (Algorithm 2) for finding an approximate FOSP of (1) without requiring any prior knowledge of these parameters. Finally, we propose a Newton-CG method (Algorithm 3) to find an approximate SOSP of (1). By leveraging a novel inexact oracle (see Lemma 1), we show that these methods achieve the best-known iteration and operation complexity for finding an approximate FOSP or SOSP of (1). In addition, preliminary numerical results are presented, demonstrating the practical advantages of our parameter-free Newton-CG method over the cubic regularized Newton method [16].

The main contributions of this paper are as follows.

- We propose Newton-CG methods (Algorithms 1 and 3) to find an approximate FOSP and SOSP of (1), respectively, assuming that the parameters associated with the Hölder continuity of $\nabla^2 f$ are explicitly known. In contrast with the regularized Newton methods [16, 31], our methods solve much simpler subproblems, while achieving the best-known iteration and operation complexity.

- We propose a *parameter-free* Newton-CG method (Algorithm 2) for finding an approximate FOSP of (1) without requiring prior knowledge of these parameters. To the best of our knowledge, this is *the first parameter-free second-order method* for finding an approximate FOSP of (1), while achieving the best-known iteration and operation complexity.

[1] $\widetilde{\mathcal{O}}(\cdot)$ represents $\mathcal{O}(\cdot)$ with logarithmic terms omitted.

- We introduce a novel inexact oracle (see Lemma 1) as the framework for the design and analysis of our Newton-CG methods. It substantially facilitates our development and analysis and shall be a useful tool for further algorithmic development for problem (1).

The remainder of this paper is organized as follows. In Section 2, we introduce some notation and make some assumptions on the problem studied in this paper. In Section 3, we propose a Newton-CG method for finding an approximate FOSP of (1) and study its complexity. In Section 4, we propose a parameter-free Newton-CG method for finding an approximate FOSP of (1) and study its complexity. In Section 5, we propose a Newton-CG method for finding an approximate SOSP of (1) and study its complexity. Section 6 presents preliminary numerical results. In Section 7, we present the proofs of the main results. Finally, we discuss some future research directions in Section 8.

2 Notation and assumptions

Throughout this paper, we let $\mathbb{R}^n$ denote the n-dimensional Euclidean space. We use $\|\cdot\|$ to denote the Euclidean norm of a vector or the spectral norm of a matrix. For any $s \in \mathbb{R}$, we let s_+ and $\lceil s \rceil$ denote the nonnegative part of s and the least integer no less than s, respectively, and we let $\mathrm{sgn}(s)$ be 1 if $s \geq 0$ and let it be -1 otherwise. For a real symmetric matrix H, we use $\lambda_{\min}(H)$ to denote its minimum eigenvalue. For any bounded set $\mathcal{S}$, we let $D_{\mathcal{S}}$ be the diameter of $\mathcal{S}$, that is, $D_{\mathcal{S}} = \sup_{x,y\in\mathcal{S}} \|x-y\|$. In addition, $\widetilde{\mathcal{O}}(\cdot)$ represents $\mathcal{O}(\cdot)$ with logarithmic terms omitted.

We make the following assumptions on problem (1) throughout this paper.

Assumption 1. (a) *The level set $\mathscr{L}_f(x^0) := \{x : f(x) \leq f(x^0)\}$ is compact for some $x^0 \in \mathbb{R}^n$.*

(b) *The function $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable, and $\nabla^2 f$ is Hölder continuous in a bounded convex open neighborhood, denoted by Ω, of $\mathscr{L}_f(x^0)$, i.e., there exist $\nu \in [0,1]$ and a finite $H_\nu > 0$ such that*

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq H_\nu \|x-y\|^\nu, \quad \forall x,y \in \Omega. \tag{3}$$

It follows from Assumption 1(a) that there exist $f_{\mathrm{low}} \in \mathbb{R}$, $U_g > 0$ and $U_H > 0$ such that

$$f(x) \geq f_{\mathrm{low}}, \quad \|\nabla f(x)\| \leq U_g, \quad \|\nabla^2 f(x)\| \leq U_H, \quad \forall x \in \mathscr{L}_f(x^0). \tag{4}$$

We now make some remarks about Assumption 1(b).

Remark 1. (i) *Assumption 1(b) includes a large class of smoothness conditions of $\nabla^2 f$. Indeed, when $\nu = 1$, the condition* (3) *recovers the standard Lipschitz continuity of $\nabla^2 f$. When $\nu = 0$, the condition* (3) *means that variations of $\nabla^2 f$ on Ω are bounded, which is equivalent to the boundedness of $\nabla^2 f$ on Ω due to the boundedness of Ω imposed in Assumption 1(b). Moreover, when $\nu \in (0,1]$, the condition* (3) *implies that $\nabla^2 f$ is uniformly continuous on Ω.*

(ii) *When the value of ν in Assumption 1(b) is larger, the smoothness of the Hessian of f is stronger. Indeed, we let $0 \leq \nu_1 < \nu_2 \leq 1$ and suppose that $\nabla^2 f$ is Hölder continuous with $\nu_2 \in [0,1]$ and a finite $H_{\nu_2} > 0$. Since Ω is bounded, it follows that*

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq H_{\nu_2}\|x-y\|^{\nu_2} \leq H_{\nu_2} D_\Omega^{\nu_2-\nu_1}\|x-y\|^{\nu_1}.$$

Hence, $\nabla^2 f$ is Hölder continuous with $\nu_1 \in [0,1]$ and $H_{\nu_1} = H_{\nu_2} D_\Omega^{\nu_2-\nu_1} > 0$.

(iii) *As a direct consequence of Assumption 1(b), one can verify that the two descent inequalities below hold for all $x, y \in \Omega$ (e.g., see equations (2.7) and (2.8) in [16]):*

$$\|\nabla f(y) - \nabla f(x) - \nabla^2 f(x)(y-x)\| \leq \frac{H_\nu \|y-x\|^{1+\nu}}{1+\nu}, \tag{5}$$

$$f(y) \leq f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(x)(y-x) + \frac{H_\nu\|y-x\|^{2+\nu}}{(1+\nu)(2+\nu)}. \tag{6}$$

Let us introduce a class of functions that satisfy Assumption 1(b) with $\nu \in (0,1]$.

Example 1. *Consider a function $f(x) = \phi(x)_+^{2+\nu}$, where $\phi : \mathbb{R}^n \to \mathbb{R}$ is twice Lipschitz continuously differentiable in a compact convex set $\mathcal{S}$, that is, there exists $L_H^\phi > 0$ such that*

$$\|\nabla^2 \phi(x) - \nabla^2 \phi(y)\| \leq L_H^\phi \|x - y\|, \qquad \forall x, y \in \mathcal{S}\,.$$

By the definition of f, one can verify that

$$\nabla^2 f(x) = (2+\nu)\phi(x)_+^{1+\nu} \nabla^2 \phi(x) + (1+\nu)(2+\nu)\phi(x)_+^\nu \nabla \phi(x) \nabla \phi(x)^T.$$

Here, observe that $\phi(x)_+^{1+\nu}$ is Lipschitz continuous in $\mathcal{S}$. Also, recall that for all $a, b \in \mathbb{R}$, we have $a_+^\nu - b_+^\nu \leq |a-b|^\nu$, which implies that $\phi(\cdot)_+^\nu$ is Hölder continuous in $\mathcal{S}$ with $\nu \in (0,1]$. In view of these and the fact that $\nabla^2 \phi$ is Lipschitz continuous in $\mathcal{S}$, we conclude that $\nabla^2 f$ is Hölder continuous in $\mathcal{S}$ with $\nu \in (0,1]$.

3 A Newton-CG method for seeking an FOSP

In this section, we propose a Newton-CG method in Algorithm 1 for seeking an ϵ-FOSP of problem (1) that satisfies $\|\nabla f(x)\| \leq \epsilon$, and then analyze its complexity results.

We first review a modified CG method, referred to as *capped CG method*, that will be used in Algorithm 1. The capped CG method was proposed in [28, Algorithm 1] for solving a possibly indefinite linear system

$$(H + 2\varepsilon I)d = -g, \tag{7}$$

where $0 \neq g \in \mathbb{R}^n$, $\varepsilon > 0$, and $H \in \mathbb{R}^{n \times n}$ is a symmetric matrix. The capped CG method terminates within a finite number of iterations and returns either an approximate solution d to (7) satisfying $\|(H+2\varepsilon I)d + g\| \leq \hat{\zeta}\|g\|$ and $d^T H d \geq -\varepsilon \|d\|^2$ for some $\hat{\zeta} \in (0,1)$ or a sufficiently negative curvature direction d of H with $d^T H d < -\varepsilon \|d\|^2$. For ease of reference, we present the capped CG method in Algorithm 4 in Appendix A.

We now introduce our Newton-CG method (Algorithm 1) for solving problem (1). At each iteration k, if the current iterate x^k does not satisfy $\|\nabla f(x^k)\| \leq \epsilon$, the capped CG method (Algorithm 4) is invoked to find either an inexact Newton direction or a negative curvature direction by solving a damped Newton system of the form:

$$(\nabla^2 f(x^k) + 2\sqrt{\gamma_\nu(\epsilon)\epsilon} I)d = -\nabla f(x^k), \tag{8}$$

where $\gamma_\nu(\cdot)$ is the inexact Lipschitz continuity modulus [2] defined as

$$\gamma_\nu(\epsilon) := 4 H_\nu^{\frac{2}{1+\nu}} \epsilon^{-\frac{1-\nu}{1+\nu}}. \tag{9}$$

The search direction d^k and step length α_k are then produced, depending on the type of d^k, and the next iterate x^{k+1} is generated based on d^k and α_k. Details of this Newton-CG method are presented in Algorithm 1.

Before analyzing Algorithm 1, we make some remarks about the damped Newton system (8). Notice that directly applying a conjugate gradient (CG) method to an indefinite Newton system, associated with problem (1), may not produce a sufficiently descent direction for the objective f. To overcome this issue, the authors of [28] proposed to solve a slightly damped Newton system:

$$\left(\nabla^2 f(x^k) + 2\sqrt{\epsilon} I\right) d = -\nabla f(x^k),^{3} \tag{10}$$

where $\sqrt{\epsilon}$ is the damping parameter. Leveraging this idea, they developed a Newton-CG method that achieves an iteration complexity of $\mathcal{O}(L_H^3 \epsilon^{-3/2})$ for finding an ϵ-FOSP of nonconvex unconstrained optimization problems, where L_H is the Lipschitz continuity modulus. However, the dependence on L_H in this complexity result can be improved. It can be verified that by using the line search techniques developed in [17, Algorithm 1] and

[2] The inexact Lipschitz continuity modulus has been widely used to study first-order methods with Hölder continuous gradient (e.g., see [15, 18, 25]).

[3] The damping parameter is set as ϵ_H in [28], where ϵ_H is the tolerance for the second-order optimality condition. It was also mentioned in [28, Section 4.2] that to achieve the iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$ for finding an ϵ-FOSP, one should choose $\epsilon_H^2 = \epsilon$.

replacing the damping parameter $\sqrt{\epsilon}$ in (10) with $\sqrt{L_H\epsilon}$, the resulting iteration complexity can be improved to $\mathcal{O}(L_H^{1/2}\epsilon^{-3/2})$, matching the best-known result stated in (2) with $\nu = 1$. Based on this observation, for the Hölder continuous case, we propose to set the damping parameter as $\sqrt{\gamma_\nu(\epsilon)\epsilon}$ with $\gamma_\nu(\epsilon)$ being the inexact Lipschitz continuity modulus. We next show in Theorem 2 and Remark 2 that with this choice of damping parameter, our proposed Newton-CG method achieves the optimal iteration complexity of second-order methods for finding an ϵ-FOSP of problem (1) under Assumption 1.

Algorithm 1 A Newton-CG method for finding an ϵ-FOSP of (1)

```
input: tolerance $\epsilon \in (0,1)$, starting point $x^0$, CG-accuracy parameter $\zeta \in (0,1)$, $\gamma_\nu(\epsilon)$ given in (9);
Set $k \leftarrow 0$;
while $\|\nabla f(x^k)\| > \epsilon$ do
    Call Algorithm 4 (Appendix A) with $H = \nabla^2 f(x^k)$, $\varepsilon = \sqrt{\gamma_\nu(\epsilon)\epsilon}$, $g = \nabla f(x^k)$, accuracy parameter $\zeta$ and $U = 0$ to obtain outputs $d$, d_type;
    if d_type=NC then
        Set
            $$d^k = -\operatorname{sgn}(d^T\nabla f(x^k))\frac{|d^T\nabla^2 f(x^k)d|}{\|d\|^3}d \quad \text{and} \quad \alpha_k = 1/\gamma_\nu(\epsilon); \tag{11}$$
    else {d_type=SOL}
        Set
            $$d^k = d \quad \text{and} \quad \alpha_k = \min\left\{1, \frac{[\epsilon/\gamma_\nu(\epsilon)]^{1/4}}{2\|d\|^{1/2}}\right\}; \tag{12}$$
    end if
    Set $x^{k+1} = x^k + \alpha_k d^k$ and $k \leftarrow k+1$;
end while
```

The following theorem shows that f is nonincreasing along the iterates generated by Algorithm 1. Its proof is deferred to Section 7.1.

Theorem 1 (monotonicity of Algorithm 1). *Suppose that Assumption 1 holds. Let $\{x^k\}_{k\in\mathbb{K}_1}$ be all the iterates generated by Algorithm 1, where $\mathbb{K}_1$ is a subset of consecutive nonnegative integers starting from 0. Then $\{f(x^k)\}_{k\in\mathbb{K}_1}$ is nonincreasing.*

The following theorem states the iteration and operation complexity of Algorithm 1, whose proof is relegated to Section 7.1.

Theorem 2 (iteration and operation complexity of Algorithm 1). *Suppose that Assumption 1 holds. Let*

$$K_1 = \left\lceil 144(f(x^0) - f_{\text{low}})\gamma_\nu(\epsilon)^{1/2}\epsilon^{-3/2}\right\rceil + 1, \tag{13}$$

where f_{low} and $\gamma_\nu(\epsilon)$ are given in (4) and (9), respectively. Then the following statements hold.

(i) *Algorithm 1 terminates in at most K_1 iterations.*

(ii) *The total main operations of Algorithm 1 consist of*

$$\widetilde{\mathcal{O}}\left(K_1 \min\left\{n, [\gamma_\nu(\epsilon)\epsilon]^{-1/4}\right\}\right)$$

gradient evaluations and Hessian-vector products of f.

Remark 2. *From Theorem 2, we observe that Algorithm 1 achieves an iteration and operation complexity of*

$$\mathcal{O}\left(H_\nu^{\frac{1}{1+\nu}}\epsilon^{-\frac{2+\nu}{1+\nu}}\right) \quad \textit{and} \quad \widetilde{\mathcal{O}}\left(H_\nu^{\frac{1}{1+\nu}}\epsilon^{-\frac{2+\nu}{1+\nu}}\min\left\{n, (H_\nu\epsilon^\nu)^{-\frac{1}{2(1+\nu)}}\right\}\right) \tag{14}$$

for finding an ϵ-FOSP of problem (1), respectively. In particular, the iteration complexity in (14) has been shown to be optimal in [11], and has also been achieved by the cubic regularized Newton method in [16]. When $\nu = 1$, the complexity results in (14) recover the iteration and operation complexity of $\mathcal{O}(\epsilon^{-3/2})$ and $\widetilde{\mathcal{O}}(\epsilon^{-3/2}\min\{n, \epsilon^{-1/4}\})$, respectively, established for the Newton-CG method in [28] for finding an ϵ-FOSP, which are the best-known results for second-order methods.

4 A parameter-free Newton-CG method for seeking an FOSP

After the previous discussions, we can observe that Algorithm 1 achieves the best-known iteration complexity for finding an ϵ-FOSP, and its fundamental operations rely only on gradient evaluations and Hessian-vector products of f. Nonetheless, computing the parameter $\gamma_\nu(\epsilon)$ given in (9) still requires knowing the problem parameters ν and H_ν associated with the Hölder continuity of $\nabla^2 f$. These parameters may not be available for a sophisticated function f. Even if known, these parameters are not unique. The tighter value of them typically leads to a faster convergent algorithm. Yet, it may be challenging to find the tightest possible value for them. In light of these challenges, we next propose a parameter-free Newton-CG method in Algorithm 2, equipped with an innovative backtracking scheme for estimating the inexact Lipschitz continuity modulus $\gamma_\nu(\epsilon)$. This method enjoys the same order of iteration and operation complexity guarantees as Algorithm 1 for finding an ϵ-FOSP of (1) without prior knowledge of ν and H_ν.

Algorithm 2 A parameter-free Newton-CG method for finding an ϵ-FOSP of (1)

```
input: tolerance $\epsilon \in (0,1)$, starting point $x^0$, CG-accuracy parameter $\zeta \in (0,1)$, trial parameter $\gamma_{-1} > 0$, backtracking ratio $\theta > 1$;
Set $k \leftarrow 0$;
while $\|\nabla f(x^k)\| > \epsilon$ do
    Set $\tilde{x} = x^k$ and $\tilde{\gamma} = \max\{\gamma_{-1}, \gamma_{k-1}/\theta\}$;
    for $t = 0, 1, 2, \ldots$ do
        Set $\tilde{\gamma}_t = \theta^t \tilde{\gamma}$;
        Call Algorithm 4 (Appendix A) with $H = \nabla^2 f(\tilde{x})$, $\varepsilon = \sqrt{\tilde{\gamma}_t \epsilon}$, $g = \nabla f(\tilde{x})$, accuracy parameter $\zeta$, and $U = 0$ to obtain
        outputs $d$, d_type;
        if d_type=NC then
            Set
                $\tilde{d}^t = -\operatorname{sgn}(d^T \nabla f(\tilde{x})) \frac{|d^T \nabla^2 f(\tilde{x}) d|}{\|d\|^3} d$   and   $\tilde{\alpha}_t = 1/\tilde{\gamma}_t$;        (15)
            Break the inner loop if $\tilde{\alpha}_t$ and $\tilde{d}^t$ satisfy
                $f(\tilde{x} + \tilde{\alpha}_t \tilde{d}^t) \le f(\tilde{x}) - \tilde{\alpha}_t^2 \|\tilde{d}^t\|^3/6.$        (16)
        else {d_type=SOL}
            Set
                $\tilde{d}^t = d$   and   $\tilde{\alpha}_t = \min\left\{1, \frac{(\epsilon/\tilde{\gamma}_t)^{1/4}}{2\|d\|^{1/2}}\right\}$;        (17)
            Break the inner loop if $\tilde{\alpha}_t$ and $\tilde{d}^t$ satisfy
                $\|\nabla f(\tilde{x} + \tilde{\alpha}_t \tilde{d}^t)\| \le \epsilon, \quad f(\tilde{x} + \tilde{\alpha}_t \tilde{d}^t) \le f(\tilde{x}).$        (18)
            if $\tilde{\alpha}_t < 1$ then
                Break the inner loop if $\tilde{\alpha}_t$ and $\tilde{d}^t$ satisfy
                    $f(\tilde{x} + \tilde{\alpha}_t \tilde{d}^t) \le f(\tilde{x}) - \sqrt{\tilde{\gamma}_t \epsilon}\, \tilde{\alpha}_t^2 \|\tilde{d}^t\|^2/2.$        (19)
            else {$\tilde{\alpha}_t = 1$}
                Break the inner loop if (19) holds and $\tilde{d}^t$ satisfies
                    $\|\nabla f(\tilde{x} + \tilde{d}^t) - \nabla f(\tilde{x}) - \nabla^2 f(\tilde{x}) \tilde{d}^t\| \le 2\tilde{\gamma}_t \|\tilde{d}^t\|^2 + \epsilon/2.$        (20)
            end if
        end if
    end for
    Set $d^k = \tilde{d}^t$, $\alpha_k = \tilde{\alpha}_t$, and $\gamma_k = \tilde{\gamma}_t$;
    Set $x^{k+1} = x^k + \alpha_k d^k$ and $k \leftarrow k + 1$;
end while
```

We now describe the parameter-free Newton-CG method (Algorithm 2) for solving (1). At each iteration k, the capped CG method (Algorithm 4) is invoked to solve a damped Newton system (8) with the parameter $\gamma_\nu(\epsilon)$ replaced by a trial value $\tilde{\gamma}_t$. This gives a trial search direction $\tilde{d}^t$ and a trial step length $\tilde{\alpha}_t$. Next, this method checks whether an ϵ-FOSP of (1) is found (see (18)), or whether $(\tilde{\gamma}_t, \tilde{\alpha}_t, \tilde{d}^t)$ satisfies certain condition that ensure sufficient reduction for f (see (16), (19), and (20)). If not, this method increases the trial value $\tilde{\gamma}_t$ by a ratio $\theta > 1$, and repeats the above process. Otherwise, this method breaks the inner loop and updates the next iterate as $x^{k+1} = x^k + \alpha_k d^k$ with $d^k = \tilde{d}^t$ and $\alpha_k = \tilde{\alpha}_t$. The detailed description of this parameter-free Newton-CG method is presented in Algorithm 2.

The following theorem shows that the number of calls of Algorithm 4 at each iteration of Algorithm 2 is finite, and therefore, Algorithm 2 is well-defined. Its proof is deferred to Section 7.2.

Theorem 3 (**well-definedness of Algorithm 2**). *Suppose that Assumption 1 holds and that x^k satisfying $\|\nabla f(x^k)\| > \epsilon$ is generated by Algorithm 2. Let*

$$T := |\log(\gamma_\nu(\epsilon)/\gamma_{-1})/\log\theta| + 1, \qquad \bar{\gamma}_\nu(\epsilon) := \max\{\gamma_{-1}, \theta\gamma_\nu(\epsilon)\}, \tag{21}$$

where $\gamma_\nu(\epsilon)$ is defined as in (9)*, and γ_{-1} and θ are inputs of Algorithm 2. Then the number of calls of Algorithm 4 at iteration k of Algorithm 2 is bounded above by T. Moreover, $\gamma_k \leq \bar{\gamma}_\nu(\epsilon)$.*

The next theorem states the iteration and operation complexity of Algorithm 2. Its proof is deferred to Section 7.2.

Theorem 4 (**iteration and operation complexity of Algorithm 2**). *Suppose that Assumption 1 holds. Let*

$$K_2 = \left\lceil 72(f(x^0) - f_{\text{low}})[\bar{\gamma}_\nu(\epsilon)]^{1/2}\epsilon^{-3/2} \right\rceil + 1, \tag{22}$$

where f_{low} and $\bar{\gamma}_\nu(\epsilon)$ are given in (4) *and* (21)*, respectively. Then the following statements hold.*

(i) *Algorithm 2 terminates in at most K_2 iterations.*

(ii) *The total main operations of Algorithm 2 consist of*

$$\widetilde{\mathcal{O}}\left(TK_2 \min\left\{n, \epsilon^{-1}K_2^{-1/2}\right\}\right)$$

gradient evaluations and Hessian-vector products of f, where T is defined in (21)*.*

Remark 3. *From Theorem 4, we see that Algorithm 2 achieves an iteration and operation complexity of*

$$\mathcal{O}\left(H_\nu^{\frac{1}{1+\nu}}\epsilon^{-\frac{2+\nu}{1+\nu}}\right) \quad \text{and} \quad \widetilde{\mathcal{O}}\left(H_\nu^{\frac{1}{1+\nu}}\epsilon^{-\frac{2+\nu}{1+\nu}}\min\left\{n, (H_\nu\epsilon^\nu)^{-\frac{1}{2(1+\nu)}}\right\}\right) \tag{23}$$

for finding an ϵ-FOSP of problem (1)*, respectively. In particular, the iteration complexity in* (23) *has been shown to be optimal in [11] and has also been achieved by the cubic regularized Newton method in [16]. Algorithm 2 is the first parameter-free second-order method that achieves the best-known iteration and operation complexity for finding an ϵ-FOSP of nonconvex unconstrained optimization problems with Hölder continuous Hessians. When $\nu = 1$, the complexity results in* (23) *recover the iteration and operation complexity of $\mathcal{O}(\epsilon^{-3/2})$ and $\widetilde{\mathcal{O}}(\epsilon^{-3/2}\min\{n, \epsilon^{-1/4}\})$, respectively, established for the Newton-CG method in [28] for finding an ϵ-FOSP, which are the best-known results for second-order methods.*

5 A Newton-CG method for seeking an SOSP

In this section we propose a Newton-CG method in Algorithm 3 for seeking an (ϵ_g, ϵ_H)-SOSP of problem (1) that satisfies

$$\|\nabla f(x)\| \leq \epsilon_g, \qquad \lambda_{\min}(\nabla^2 f(x)) \geq -\epsilon_H,$$

where $\epsilon_g, \epsilon_H \in (0, 1)$ are tolerances. We also establish the iteration and operation complexity of this algorithm under Assumption 1 with $\nu \in (0, 1]$.

We first review the minimum eigenvalue oracle that will be used in Algorithm 3. This oracle was proposed in [28] to check whether a direction of sufficiently negative curvature exists for a given symmetric matrix H. It either produces a sufficiently negative curvature direction v of H with $\|v\| = 1$ and $v^T Hv \leq -\epsilon_H/2$ or certifies that $\lambda_{\min}(H) \geq -\epsilon_H$ holds with high probability. For ease of reference, we present the minimum eigenvalue oracle in Algorithm 3 in Appendix B.

We now describe the Newton-CG method (Algorithm 3) for seeking an (ϵ_g, ϵ_H)-SOSP of problem (1). At each iteration k, this algorithm starts by checking whether the current iterate x^k satisfies $\|\nabla f(x^k)\| \le \epsilon_g$. If not, then this algorithm updates the next iterate x^{k+1} in the same manner as Algorithm 1 with ϵ replaced by ϵ_g. Specifically, the capped CG method (Algorithm 4) is applied to the damped Newton system (8) with ϵ replaced by ϵ_g to obtain an inexact Newton direction or a sufficiently negative curvature direction, and the next iterate x^{k+1} is generated accordingly. Otherwise, if $\|\nabla f(x^k)\| > \epsilon_g$, the minimum eigenvalue oracle (Algorithm 5) is invoked to check whether a direction of sufficiently negative curvature exists for the Hessian $\nabla^2 f(x^k)$. Specifically, this oracle either produces a sufficiently negative curvature direction of $\nabla^2 f(x^k)$ and computes the next iterate x^{k+1}, or certifies that the minimum eigenvalue of $\nabla^2 f(x^k)$ is larger than $-\epsilon_H$ and terminates this algorithm.

Algorithm 3 A Newton-CG method for seeking an (ϵ_g, ϵ_H)-SOSP of (1)

```
input: tolerances $\epsilon_g, \epsilon_H \in (0,1)$, starting point $x^0$, CG-accuracy parameter $\zeta \in (0,1)$, $\gamma_\nu(\epsilon_g)$ given in (9);
for $k = 0, 1, 2, \ldots$ do
  if $\|\nabla f(x^k)\| > \epsilon_g$ then
    Call Algorithm 4 (Appendix A) with $H = \nabla^2 f(x^k)$, $\varepsilon = \sqrt{\gamma_\nu(\epsilon_g)\epsilon_g}$, $g = \nabla f(x^k)$, accuracy parameter $\zeta$, and
    $U = 0$ to obtain outputs $d$, d_type;
    if d_type=NC then
      Set
        $$d^k = -\operatorname{sgn}(d^T \nabla f(x^k)) \frac{|d^T \nabla^2 f(x^k) d|}{\|d\|^3} d \quad \text{and} \quad \alpha_k = 1/\gamma_\nu(\epsilon_g); \qquad (24)$$
    else {d_type=SOL}
      Set
        $$d^k = d \quad \text{and} \quad \alpha_k = \min\left\{1, \frac{[\epsilon_g/\gamma_\nu(\epsilon_g)]^{1/4}}{2\|d\|^{1/2}}\right\}; \qquad (25)$$
    end if
  else
    Call Algorithm 5 (Appendix B) with $H = \nabla^2 f(x^k)$ and $\varepsilon = \epsilon_H$, and probability parameter $\delta$;
    if Algorithm 5 certifies that $\lambda_{\min}(\nabla^2 f(x^k)) \ge -\epsilon_H$ then
      Output $x^k$ and terminates;
    else {Sufficiently negative curvature direction $v$ returned by Algorithm 5}
      Set
        $$d^k = -\operatorname{sgn}(v^T \nabla f(x^k)) |v^T \nabla^2 f(x^k) v| v \quad \text{and} \quad \alpha_k = (\epsilon_H/2)^{(1-\nu)/\nu}/(2H_\nu)^{1/\nu}; \qquad (26)$$
    end if
  end if
  Set $x^{k+1} = x^k + \alpha_k d^k$;
end for
```

The following lemma shows that f is nonincreasing along the iterates generated by Algorithm 3, whose proof is deferred to Section 7.3.

Theorem 5 (monotonicity of Algorithm 3). *Suppose that Assumption 1 holds with $\nu \in (0,1]$. Let $\{x^k\}_{k\in\mathbb{K}_3}$ be all the iterates generated by Algorithm 3, where $\mathbb{K}_3$ is a subset of consecutive nonnegative integers starting from 0. Then $\{f(x^k)\}_{k\in\mathbb{K}_3}$ is nonincreasing.*

The following theorem states the iteration and operation complexity of Algorithm 3, whose proof is relegated to Section 7.3.

Theorem 6 (complexity of Algorithm 3). *Suppose that Assumption 1 holds with $\nu \in (0,1]$. Let*

$$\widetilde{K}_1 = \left\lceil 144(f(x^0) - f_{\text{low}})[\gamma_\nu(\epsilon_g)]^{1/2}\epsilon_g^{-3/2}\right\rceil + \left\lceil 4(f(x^0) - f_{\text{low}})(\epsilon_H/2)^{-(2+\nu)/\nu}/(2H_\nu)^{2/\nu}\right\rceil + 1, \qquad (27)$$

$$\widetilde{K}_2 = \left\lceil 4(f(x^0) - f_{\text{low}})(\epsilon_H/2)^{-(2+\nu)/\nu}/(2H_\nu)^{2/\nu}\right\rceil + 1, \qquad (28)$$

where f_{low} and $\gamma_\nu(\cdot)$ are defined in (4) and (9), respectively. Then the following statements hold.

(i) *The total number of calls of Algorithm 5 is at most $\widetilde{K}_2$.*

(ii) *The total number of calls of Algorithm 4 is at most* $\widetilde{K}_1$.

(iii) *Algorithm 3 terminates in at most* $\widetilde{K}_1+\widetilde{K}_2$ *iterations. Its output* x^k *satisfies* $\|\nabla f(x^k)\| \le \epsilon_g$ *deterministically for some* $k \le \widetilde{K}_1 + \widetilde{K}_2$. *Moreover, it satisfies* $\lambda_{\min}(\nabla^2 f(x^k)) \ge -\epsilon_H$ *with probability at least* $1-\delta$.

(iv) *The total main operations of Algorithm 3 consist of*

$$\widetilde{\mathcal{O}}\left(\min\left\{n, [\gamma_\nu(\epsilon_g)\epsilon_g]^{-1/4}\right\}\widetilde{K}_1 + \min\left\{n, \epsilon_H^{-1/2}\right\}\widetilde{K}_2\right)$$

gradient evaluations and Hessian-vector products of f.

Remark 4. *From Theorem 6, we observe that Algorithm 3 achieves an iteration and operation complexity of*

$$\mathcal{O}\left(H_\nu^{\frac{1}{1+\nu}}\epsilon_g^{-\frac{2+\nu}{1+\nu}} + H_\nu^{\frac{2}{\nu}}\epsilon_H^{-\frac{2+\nu}{\nu}}\right) \quad \text{and} \tag{29}$$

$$\widetilde{\mathcal{O}}\left(\left(H_\nu^{\frac{1}{1+\nu}}\epsilon_g^{-\frac{2+\nu}{1+\nu}} + H_\nu^{\frac{2}{\nu}}\epsilon_H^{-\frac{2+\nu}{\nu}}\right)\min\left\{n, (H_\nu\epsilon_g^{\nu})^{-\frac{1}{2(1+\nu)}}\right\} + H_\nu^{\frac{2}{\nu}}\epsilon_H^{-\frac{2+\nu}{\nu}}\min\left\{n, \epsilon_H^{-\frac{1}{2}}\right\}\right) \tag{30}$$

for finding an (ϵ_g,ϵ_H)*-SOSP of problem* (1) *with high probability, respectively. When* $\nu=1$, *the iteration and operation complexity results in* (29) *and* (30) *reduce to* $\mathcal{O}(\epsilon_g^{-3/2}+\epsilon_H^{-3})$ *and* $\widetilde{\mathcal{O}}((\epsilon_g^{-3/2}+\epsilon_H^{-3})\min\{n,\epsilon_g^{-1/4}\} + \epsilon_H^{-3}\min\{n,\epsilon_H^{-1/2}\})$, *respectively, which retain or improve the complexity results of the Newton-CG method in [28] for finding an* (ϵ_g,ϵ_H)*-SOSP.*

6 Numerical results

In this section, we conduct some preliminary numerical experiments to test the performance of our parameter-free Newton-CG method (Algorithm 2) and the cubic regularized Newton method with line search (Universal Method II) in [16]. All the algorithms are coded in Matlab and all the computations are performed on a desktop with a 3.79 GHz AMD 3900XT 12-Core processor and 32 GB of RAM.

6.1 Infeasibility detection

In this subsection, we consider the problem of infeasibility detection (see [5]):

$$\min_{x\in\mathbb{R}^n} \sum_{i=1}^{m} \left(x^T A_i x + b_i^T x + c_i\right)_+^p, \tag{31}$$

where $p > 2$, $A_i \in \mathbb{R}^{n\times n}$, $b_i \in \mathbb{R}^n$, and $c_i \in \mathbb{R}$ for $1 \le i \le m$.

For each pair (n, m, p), we randomly generate 10 instances of problem (31). In particular, we first randomly generate $A_i = U_i D_i U_i^T$, $1 \le i \le m$, where the D_i is a randomly generated diagonal matrix, and U_i is a randomly generated orthogonal matrix. Each diagonal element of D_i, $1 \le i \le n$, is uniformly distributed over $[-1, n-1]$. We then randomly generate b_i, $1 \le i \le m$, with each component according to the uniform distribution over $[0, n]$, and fix $c_i = 1$ for $1 \le i \le m$.

Our aim is to find a 10^{-4}-FOSP of problem (31) for the above instances by Algorithm 2 and the cubic regularized Newton method with line search in [16] and compare their performance. For the latter method, we adopt the approach proposed in [26] to solve its cubic regularized subproblems. For both methods, we choose the initial point as $x^0 = (0,\ldots,0)^T$, and the other parameters as

- $(\zeta, \gamma_{-1}, \theta) = (0.5, 10, 2)$ for Algorithm 2;
- $H_0 = 10$ for the cubic regularized Newton method ([16]).

The computational results of Algorithm 2 and the cubic regularized Newton method in [16] (abbreviated as CRN-LS) for solving problem (31) for the instances randomly generated above are presented in Table 1. In detail, the values of n, m, and p are listed in the first two columns, respectively. For each triple (n, m, p), the

			Objective value		CPU time (seconds)		Total subproblems	
n	m	p	Algorithm 2	CRN-LS	Algorithm 2	CRN-LS	Algorithm 2	CRN-LS
100	2	2.25	4.5×10^{-14}	1.4×10^{-14}	1.1	4.7	163.1	168.0
100	2	2.5	3.9×10^{-13}	1.5×10^{-13}	0.93	4.8	142.2	185.3
100	2	2.75	1.8×10^{-12}	1.5×10^{-12}	0.89	4.4	125.7	197.2
100	2	3	6.8×10^{-12}	3.7×10^{-12}	0.90	4.6	112.9	206.7
300	6	2.25	8.4×10^{-16}	1.1×10^{-15}	15.6	68.8	221.9	348.2
300	6	2.5	1.2×10^{-14}	1.9×10^{-15}	15.5	69.9	185.0	384.5
300	6	2.75	7.8×10^{-14}	4.1×10^{-14}	15.5	68.6	168.2	400.0
300	6	3.0	3.3×10^{-13}	1.7×10^{-13}	15.1	68.0	153.7	418.0
500	10	2.25	1.7×10^{-16}	5.0×10^{-17}	67.6	335.1	247.6	457.0
500	10	2.5	3.5×10^{-15}	3.1×10^{-15}	66.6	327.9	210.4	494.5
500	10	2.75	2.1×10^{-14}	6.7×10^{-15}	66.7	338.1	191.3	517.0
500	10	3	9.7×10^{-14}	3.6×10^{-14}	64.0	328.5	179.5	539.0

Table 1: Numerical results for problem (31)

			Objective value		CPU time (seconds)		Total subproblems	
n	m	p	Algorithm 2	CRN-LS	Algorithm 2	CRN-LS	Algorithm 2	CRN-LS
100	20	2.25	2.1	2.9	0.36	5.6	102.9	135.9
100	20	2.5	2.0	2.2	0.32	5.7	119.2	147.8
100	20	2.75	2.2	3.3	0.41	6.2	130.3	168.6
100	20	3	2.6	2.7	0.40	6.8	131.0	184.2
500	100	2.25	9.6	12.9	10.0	309.5	230.6	292.3
500	100	2.5	10.2	14.6	10.8	411.9	249.4	382.8
500	100	2.75	10.8	11.6	13.6	522.3	308.2	480.7
500	100	3	11.4	14.4	15.2	555.0	357.4	508.5
1000	200	2.25	19.3	26.2	40.3	2083.6	312.8	702.0
1000	200	2.5	20.4	25.2	56.0	2453.0	406.8	821.3
1000	200	2.75	21.2	31.8	78.3	2895.1	539.7	956.5
1000	200	3.0	22.2	22.5	84.3	3180.3	619.2	1066.2

Table 2: Numerical results for problem (32)

average final objective value, the average CPU time, and the average total number of subproblems over 10 random instances are given in the rest of the columns. Here, one subproblem refers to one cubic regularized subproblem solved by the cubic regularized method or one damped Newton system solved by Algorithm 2. One can observe that both methods output an approximate solution of a similar objective, while Algorithm 2 substantially outperforms the cubic regularized Newton method in [16] in terms of CPU time.

6.2 Single-layer neural networks

In this subsection, we consider the problem of training single-layer RePu neural networks (see [21]):

$$\min_{x\in\mathbb{R}^n} \sum_{i=1}^{m} \phi((a_i^T x)_+^p - b_i), \tag{32}$$

where $p > 2$, $\phi(t) = t^2/(1+t^2)$ is a nonconvex loss function (see [3, 7]), $a_i \in \mathbb{R}^n$, and $b_i \in \mathbb{R}$ for $1 \le i \le m$.

For each triple (n, m, p), we randomly generate 10 instances of problem (32). In particular, we first randomly generate a_i, $1 \le i \le m$, with all its components following the standard normal distribution. We then randomly generate $\bar{b}_i$, $1 \le i \le m$, according to the standard normal distribution, and set $b_i = |\bar{b}_i|$ for $1 \le i \le m$.

Our goal is to find a 10^{-4}-FOSP of problem (32) for the above instances by Algorithm 2 and the cubic regularized Newton method with line search in [16] and compare their performance. For the latter method, we adopt the approach proposed in [26] to solve its cubic regularized subproblems. For both methods, we choose the initial iterate as $x_0 = (1/n, \dots, 1/n)^T$, and set the other parameters for Algorithm 2 and the cubic regularized Newton method the same as those described in Subsection 6.1.

The computational results of Algorithm 2 and the cubic regularized Newton method in [16] for solving problem (32) for the instances randomly generated above are presented in Table 2. In detail, the value of n, m, and p are listed in the first three columns, respectively, For each triple (n, m, p), the average final objective, the average CPU time, and the average total number of subproblems over 10 random instances are given in the rest of the columns. Here, one subproblem refers to one cubic regularized subproblem solved by the cubic regularized method or one damped Newton system solved by Algorithm 2. One can observe that Algorithm 2 finds a 10^{-4}-FOSP of (32) substantially faster than the cubic regularized Newton method in [16].

7 Proof of the main results

In this section we provide a proof of our main results presented in Sections 3, 4 and 5, which are particularly Theorems 1-6.

To start with, let us establish two technical lemmas. The following lemma provides us with an inexact oracle that our analysis relies heavily on. This result is inspired by the inexact oracle introduced by Nesterov in [25] for first-order methods in solving convex optimization problems with Hölder continuous gradient.

Lemma 1. *Under Assumption 1(b), the following inequalities hold for any $\delta > 0$:*

$$\|\nabla f(y) - \nabla f(x) - \nabla^2 f(x)(y-x)\| \le \frac{1}{2} L_1(\delta)\|y-x\|^2 + \delta, \quad \forall x, y \in \Omega, \tag{33}$$

$$f(y) \le f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T \nabla^2 f(x)(y-x) + \frac{1}{3}L_2(\delta)\|y-x\|^3 + \delta, \quad \forall x, y \in \Omega. \tag{34}$$

where

$$L_1(\delta) = \left[\frac{1-\nu}{1+\nu}\cdot\frac{1}{2\delta}\right]^{\frac{1-\nu}{1+\nu}} H_\nu^{\frac{2}{1+\nu}}, \qquad L_2(\delta) = \left[\frac{1-\nu}{2+\nu}\cdot\frac{1}{3\delta}\right]^{\frac{1-\nu}{2+\nu}}\left[\frac{H_\nu}{1+\nu}\right]^{\frac{3}{2+\nu}}.[4] \tag{35}$$

Proof. The proof of (33) is identical to the one of [25, Lemma 2], and thus omitted here.

We next prove (34). When $\nu = 1$, it follows from (6) that (34) holds. We suppose for the rest of the proof that $\nu \in [0, 1)$. Recall that all $\tau, s \ge 0$ satisfy the Young's inequality

$$\tau s \le \frac{1}{p}\tau^p + \frac{1}{q}s^q,$$

where $p, q \ge 1$ and $1/p + 1/q = 1$. Taking $\tau = t^{2+\nu}$, $p = 3/(2+\nu)$, and $q = 3/(1-\nu)$, we obtain that

$$t^{2+\nu} \le \frac{2+\nu}{3s}t^3 + \frac{1-\nu}{3}s^{\frac{2+\nu}{1-\nu}}, \quad \forall t \ge 0, s > 0. \tag{36}$$

Let us denote $\delta = \frac{(1-\nu)H_\nu}{3(1+\nu)(2+\nu)}s^{\frac{2+\nu}{1-\nu}}$. Then $s = \left[\frac{3(1+\nu)(2+\nu)\delta}{(1-\nu)H_\nu}\right]^{\frac{1-\nu}{2+\nu}}$. Multiplying both sides of (36) by $\frac{H_\nu}{(1+\nu)(2+\nu)}$ and taking $t = \|y-x\|$, we obtain that

$$\frac{H_\nu\|y-x\|^{2+\nu}}{(1+\nu)(2+\nu)} \le \frac{H_\nu\|y-x\|^3}{3(1+\nu)s} + \delta \le \frac{1}{3}\left[\frac{1-\nu}{2+\nu}\cdot\frac{1}{3\delta}\right]^{\frac{1-\nu}{2+\nu}}\left[\frac{H_\nu}{1+\nu}\right]^{\frac{3}{2+\nu}}\|y-x\|^3 + \delta,$$

which along with (6) and (35) implies that (34) holds as desired. □

The following lemma provides useful properties of $L_1(\cdot)$ and $L_2(\cdot)$.

Lemma 2. *For any $c_1 \ge 2$ and $c_2 \ge 3$, we have*

$$L_1(\epsilon/c_1) \le c_1\gamma_\nu(\epsilon)/8, \tag{37}$$

$$L_2(\epsilon^{3/2}/(c_2\gamma^{1/2})) \le \sqrt{6c_2}\gamma/12, \quad \forall \gamma \ge \gamma_\nu(\epsilon), \tag{38}$$

where $L_1(\cdot)$ and $L_2(\cdot)$ are defined in (35), and $\gamma_\nu(\cdot)$ is defined in (9).

[4] By convention, 0^0 is set to 1 throughout this paper.

Proof. We first prove (37). By $c_1 \geq 2$, $\nu \in [0,1]$, (9), and (35), one has

$$L_1\left(\frac{\epsilon}{c_1}\right) = \left[\frac{1-\nu}{1+\nu}\cdot\frac{c_1}{2\epsilon}\right]^{\frac{1-\nu}{1+\nu}} H_\nu^{\frac{2}{1+\nu}} \leq \left(\frac{c_1}{2}\right)^{\frac{1-\nu}{1+\nu}} H_\nu^{\frac{2}{1+\nu}}\epsilon^{-\frac{1-\nu}{1+\nu}} \leq \frac{c_1}{2} H_\nu^{\frac{2}{1+\nu}}\epsilon^{-\frac{1-\nu}{1+\nu}} \overset{(9)}{=} \frac{c_1}{8}\gamma_\nu(\epsilon),$$

where the first equality is follows from the definition of $L_1(\cdot)$, the first inequality is due to $\nu \in [0,1]$ and $a^a \leq 1$ for all $a \in [0,1]$, and the second inequality is due to $\nu \in [0,1]$ and $c_1 \geq 2$. Hence, (37) holds as desired.

We next prove (38). By $c_2 \geq 3$, $\nu \in [0,1]$, (9), and $\gamma \geq \gamma_\nu(\epsilon)$, one has

$$\left[\frac{1-\nu}{2+\nu}\cdot\frac{c_2}{3\epsilon^{3/2}}\right]^{\frac{1-\nu}{2+\nu}}\left[\frac{H_\nu}{1+\nu}\right]^{\frac{3}{2+\nu}} \leq \sqrt{\frac{c_2}{3}} H_\nu^{\frac{3}{2+\nu}}\epsilon^{-\frac{3(1-\nu)}{2(2+\nu)}} \overset{(9)}{=} \sqrt{\frac{c_2}{3}}\left[\frac{\gamma_\nu(\epsilon)}{4}\right]^{\frac{3(1+\nu)}{2(2+\nu)}} \leq \frac{\sqrt{6}c_2}{12}\gamma^{\frac{3(1+\nu)}{2(2+\nu)}},$$

where the first inequality follows from $\nu \in [0,1]$, $c_2 \geq 3$, and $a^a \leq 1$ for all $a \in [0,1]$, and the last inequality is due to $\nu \in [0,1]$ and $\gamma \geq \gamma_\nu(\epsilon)$. Dividing both sides of this inequality by $\gamma^{(\nu-1)/[2(2+\nu)]}$ and using the definition of $L_2(\cdot)$, we obtain that

$$L_2\left(\frac{\epsilon^{3/2}}{c_2\gamma^{1/2}}\right) = \left[\frac{1-\nu}{2+\nu}\cdot\frac{c_2}{3\epsilon^{3/2}}\right]^{\frac{1-\nu}{2+\nu}}\left[\frac{H_\nu}{1+\nu}\right]^{\frac{3}{2+\nu}}\gamma^{\frac{1-\nu}{2(2+\nu)}} \leq \frac{\sqrt{6}c_2}{12}\gamma.$$

Hence, (38) holds as desired. □

7.1 Proof of the main results in Section 3

In this subsection, we first establish several technical lemmas and then use them to prove Theorems 1 and 2.

The following lemma provides some useful properties of the output of Algorithm 4, whose proof is similar to the ones of [28, Lemma 3] and [27, Lemma 7] and thus omitted here.

Lemma 3. *Suppose that Assumption 1 holds and the direction d^k results from the output d of Algorithm 4 with a type specified in d_type at some iteration k of Algorithm 1. Then the following statements hold.*

(i) *If d_type=SOL, then d^k satisfies*

$$\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\|^2 \leq (d^k)^T(\nabla^2 f(x^k) + 2\sqrt{\gamma_\nu(\epsilon)\epsilon}I)d^k, \tag{39}$$

$$(d^k)^T\nabla f(x^k) = -(d^k)^T(\nabla^2 f(x^k) + 2\sqrt{\gamma_\nu(\epsilon)\epsilon}I)d^k, \tag{40}$$

$$\|(\nabla^2 f(x^k) + 2\sqrt{\gamma_\nu(\epsilon)\epsilon}I)d^k + \nabla f(x^k)\| \leq \zeta\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\|/2. \tag{41}$$

(ii) *If d_type=NC, then d^k satisfies $(d^k)^T\nabla f(x^k) \leq 0$ and*

$$(d^k)^T\nabla^2 f(x^k)d^k/\|d^k\|^2 = -\|d^k\| \leq -\sqrt{\gamma_\nu(\epsilon)\epsilon}. \tag{42}$$

We now provide a proof of Theorem 1.

Proof of Theorem 1. We prove this theorem by induction. Let x^k, x^{k+1} be two consecutive iterates generated by Algorithm 1, and suppose that $\{f(x^\ell)\}_{0\leq\ell\leq k}$ is nonincreasing. We next prove $f(x^{k+1}) \leq f(x^k)$. Suppose for contradiction that $f(x^{k+1}) > f(x^k)$. Denote $\varphi(\alpha) = f(x^k + \alpha d^k)$. These together with $x^{k+1} = x^k + \alpha_k d^k$ imply $\varphi(\alpha_k) > \varphi(0)$. Below, we show that $\varphi(\alpha_k) > \varphi(0)$ leads to a contradiction by considering two separate cases.

Case 1) d_type=SOL. In this case, we see from Lemma 3(i) that (39)-(41) hold for d^k. Also, observe from Algorithm 1 that $\|\nabla f(x^k)\| > \epsilon$, which together with (41) implies that $d^k \neq 0$. By this, (39), and (40), one has

$$\varphi'(0) = \nabla f(x^k)^T d^k \overset{(40)}{=} -(d^k)^T(\nabla^2 f(x^k) + 2\sqrt{\gamma_\nu(\epsilon)\epsilon}I)d^k \overset{(39)}{\leq} -\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\|^2 < 0.$$

Using these and the fact that $\varphi(\alpha_k) > \varphi(0)$, we can observe that there exists a local minimizer $\alpha_* \in (0, \alpha_k)$ of φ such that $\varphi'(\alpha_*) = \nabla f(x^k + \alpha_* d^k)^T d^k = 0$ and $\varphi(\alpha_*) < \varphi(0)$, which implies that $f(x^k + \alpha_* d^k) < f(x^k) \leq f(x^0)$.

Hence, (5) holds for $x = x^k$ and $y = x^k + \alpha_* d^k$. Using this, $0 < \alpha_* < \alpha_k \le 1$, (39), (40), and $\nabla f(x^k + \alpha_* d^k)^T d^k = 0$, we deduce that

$$
\begin{aligned}
&\tfrac{\alpha_*^{1+\nu} H_\nu}{1+\nu} \|d^k\|^{2+\nu} \overset{(5)}{\ge} \|d^k\| \|\nabla f(x^k + \alpha_* d^k) - \nabla f(x^k) - \alpha_* \nabla^2 f(x^k) d^k\| \\
&\ge (d^k)^T (\nabla f(x^k + \alpha_* d^k) - \nabla f(x^k) - \alpha_* \nabla^2 f(x^k) d^k) = -(d^k)^T \nabla f(x^k) - \alpha_* (d^k)^T \nabla^2 f(x^k) d^k \\
&\overset{(40)}{=} (1 - \alpha_*)(d^k)^T (\nabla^2 f(x^k) + 2\sqrt{\gamma_\nu(\epsilon)\epsilon} I) d^k + 2\alpha_* \sqrt{\gamma_\nu(\epsilon)\epsilon} \|d^k\|^2 \overset{(39)}{\ge} (1 + \alpha_*) \sqrt{\gamma_\nu(\epsilon)\epsilon} \|d^k\|^2 \ge \sqrt{\gamma_\nu(\epsilon)\epsilon} \|d^k\|^2,
\end{aligned}
$$

which together with $d^k \ne 0$ and $\alpha_k > \alpha_*$ implies that $\alpha_k^{1+\nu} H_\nu \|d^k\|^\nu / (1 + \nu) > \sqrt{\gamma_\nu(\epsilon)\epsilon}$. By this and the definition of α_k in (12), one has

$$
\sqrt{\gamma_\nu(\epsilon)\epsilon} < \frac{\alpha_k^{1+\nu} H_\nu \|d^k\|^\nu}{1+\nu} = \frac{H_\nu}{1+\nu} \min\left\{ \|d^k\|^\nu, \frac{[\epsilon/\gamma_\nu(\epsilon)]^{\frac{1+\nu}{4}}}{2^{1+\nu} \|d^k\|^{\frac{1-\nu}{2}}} \right\} \le \frac{H_\nu}{1+\nu} \frac{[\epsilon/\gamma_\nu(\epsilon)]^{\frac{\nu}{2}}}{4^\nu} \le H_\nu \left[\frac{\epsilon}{\gamma_\nu(\epsilon)} \right]^{\frac{\nu}{2}}
$$

where the first equality follows from the definition of α_k, the second inequality is due to $\min\{a, b\} \le a^{\frac{1-\nu}{1+\nu}} b^{\frac{2\nu}{1+\nu}}$ for all $a, b > 0$, and the last inequality is due to $\nu \in [0, 1]$. Rearranging the terms of this inequality, we obtain that

$$
\gamma_\nu(\epsilon) < H_\nu^{\frac{2}{1+\nu}} \epsilon^{-\frac{1-\nu}{1+\nu}} \tag{43}
$$

which contradicts the definition of $\gamma_\nu(\epsilon)$ in (9).

Case 2) d_type=NC. In this case, we observe from Lemma 3(ii) that

$$
\nabla f(x^k)^T d^k \le 0, \quad (d^k)^T \nabla^2 f(x^k) d^k / \|d^k\|^2 = -\|d^k\| \le -\sqrt{\gamma_\nu(\epsilon)\epsilon} < 0. \tag{44}
$$

By this and the definition of φ, one has $\varphi'(0) = \nabla f(x^k)^T d^k \le 0$ and $\varphi''(0) = (d^k)^T \nabla^2 f(x^k) d^k < 0$. Using these and the fact that $\varphi(\alpha_k) > \varphi(0)$, we observe that there exists a local minimizer $\alpha_* \in (0, \alpha_k)$ of φ such that $\varphi(\alpha_*) < \varphi(0)$, namely, $f(x^k + \alpha_* d^k) < f(x^k)$. By the second-order optimality condition of φ at α_*, one has $\varphi''(\alpha_*) = (d^k)^T f(x^k + \alpha_* d^k) d^k \ge 0$. Since $f(x^k + \alpha_* d^k) < f(x^k) \le f(x^0)$, it follows that (3) holds for $x = x^k$ and $y = x^k + \alpha_* d^k$. Using this, the second relation in (44) and $(d^k)^T \nabla^2 f(x^k + \alpha_* d^k) d^k \ge 0$, we obtain that

$$
\begin{aligned}
H_\nu \alpha_*^\nu \|d^k\|^{2+\nu} &\ge \|d^k\|^2 \|\nabla^2 f(x^k + \alpha_* d^k) - \nabla^2 f(x^k)\| \ge (d^k)^T (\nabla^2 f(x^k + \alpha_* d^k) - \nabla^2 f(x^k)) d^k \\
&\ge -(d^k)^T \nabla^2 f(x^k) d^k = \|d^k\|^3.
\end{aligned} \tag{45}
$$

Recall from (44) that $\|d^k\| \ge \sqrt{\gamma_\nu(\epsilon)\epsilon} > 0$. Using this, $\alpha_k > \alpha_*$, (45), and $\alpha_k = 1/\gamma_\nu(\epsilon)$, we deduce that

$$
H_\nu / \gamma_\nu(\epsilon)^\nu = H_\nu \alpha_k^\nu \ge H_\nu \alpha_*^\nu \overset{(45)}{\ge} \|d^k\|^{1-\nu} \ge [\gamma_\nu(\epsilon)\epsilon]^{\frac{1-\nu}{2}}.
$$

Rearranging the terms of this inequality, we obtain that

$$
\gamma_\nu(\epsilon) \le H_\nu^{\frac{2}{1+\nu}} \epsilon^{-\frac{1-\nu}{1+\nu}}, \tag{46}
$$

which contradicts the definition of $\gamma_\nu(\epsilon)$ in (9).

Combining the above two cases, we conclude that $f(x^{k+1}) \le f(x^k)$, and hence $\{f(x^k)\}_{k \in \mathbb{K}_1}$ is nonincreasing. □

Our next lemma shows that when the search direction d^k in Algorithm 1 is of type 'SOL', the next iterate x^{k+1} produces a sufficient decrease in f.

Lemma 4. *Suppose that Assumption 1 holds and the direction d^k results from the output d of Algorithm 4 with d_type=SOL at some iteration k of Algorithm 1. Then $x^{k+1} = x^k + \alpha_k d^k$ satisfies either $\|\nabla f(x^{k+1})\| \le \epsilon$ or*

$$
f(x^k) - f(x^{k+1}) \ge \epsilon^{3/2} / (144 \gamma_\nu(\epsilon)^{1/2}). \tag{47}
$$

Proof. Since d_type=SOL, we see from Lemma 3(i) that (39)-(41) hold for d^k. Recall from Theorem 1 that $\{f(x^k)\}_{k\in\mathbb{K}_1}$ is nonincreasing. Thus, $f(x^k+\alpha_k d^k) = f(x^{k+1}) \le f(x^k)$. In view of this, we see that (33) and (34) hold for $y = x^k + \alpha_k d^k$ and $x = x^k$. By (34), $x^{k+1} = x^k + \alpha_k d^k$, $\alpha_k \in (0,1]$, (39), and (40), one has that for any $\delta > 0$,

$$\begin{aligned} f(x^{k+1}) - f(x^k) &\overset{(34)}{\le} \alpha_k \nabla f(x^k)^T d^k + \frac{\alpha_k^2}{2}(d^k)^T\nabla^2 f(x^k) d^k + \frac{1}{3}L_2(\delta)\alpha_k^3\|d^k\|^3 + \delta \\ &\overset{(40)}{=} -\alpha_k (d^k)^T(\nabla^2 f(x^k) + 2\sqrt{\gamma_\nu(\epsilon)\epsilon} I)d^k + \frac{\alpha_k^2}{2}(d^k)^T\nabla^2 f(x^k)d^k + \frac{1}{3} L_2(\delta)\alpha_k^3\|d^k\|^3+\delta \\ &= -\alpha_k\left(1-\frac{\alpha_k}{2}\right)(d^k)^T(\nabla^2 f(x^k)+2\sqrt{\gamma_\nu(\epsilon)\epsilon}I)d^k - \alpha_k^2\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\|^2 + \frac{1}{3} L_2(\delta)\alpha_k^3\|d^k\|^3+\delta \\ &\overset{(39)}{\le} -\alpha_k\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\|^2 + \frac{1}{3} L_2(\delta)\alpha_k^3\|d^k\|^3+\delta. \end{aligned} \tag{48}$$

Notice that if $\|\nabla f(x^{k+1})\| \le \epsilon$, the conclusion of this lemma holds. Hence, it suffices to consider the case where $\|\nabla f(x^{k+1})\| > \epsilon$. We next show that (47) holds in this case by considering two separate below.

Case 1) $\alpha_k = 1$. It follows from the definition of α_k in (12) that $\sqrt{\epsilon/\gamma_\nu(\epsilon)} \ge 4\|d^k\|$. In addition, notice from (37) with $c_1 = 2$ that $L_1(\epsilon/2) \le \gamma_\nu(\epsilon)/4$. In view these, $\|\nabla f(x^{k+1})\| > \epsilon$, (33), and (41), we have

$$\begin{aligned} \epsilon &< \|\nabla f(x^{k+1})\| = \|\nabla f(x^k + d^k)\| \\ &\le \|\nabla f(x^k+d^k) - \nabla f(x^k) - \nabla^2 f(x^k) d^k\| + \|(\nabla^2 f(x^k) + 2\sqrt{\gamma_\nu(\epsilon)\epsilon} I)d^k + \nabla f(x^k)\| + 2\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\| \\ &\overset{(33)(41)}{\le} \frac{L_1(\epsilon/2)}{2}\|d^k\|^2 + \frac{\epsilon}{2} + \frac{\zeta+4}{2}\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\| \le \left(\frac{L_1(\epsilon/2)}{8}\sqrt{\frac{\epsilon}{\gamma_\nu(\epsilon)}} + \frac{\zeta+4}{2}\sqrt{\gamma_\nu(\epsilon)\epsilon}\right)\|d^k\| + \frac{\epsilon}{2} \\ &\le \left(\frac{1}{32} + \frac{\zeta+4}{2}\right)\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\| + \frac{\epsilon}{2} \le 3\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\| + \frac{\epsilon}{2}, \end{aligned}$$

where the fourth inequality is due to $4\|d^k\| \le \sqrt{\epsilon/\gamma_\nu(\epsilon)}$, the fifth inequality is follows from $L_1(\epsilon/2) \le \gamma_\nu(\epsilon)/4$, and the last inequality is due to $\zeta \in (0,1)$. It then follows that $6\|d^k\| \ge \sqrt{\epsilon/\gamma_\nu(\epsilon)}$. Notice from (38) with $c_2 = 144$ and $\gamma = \gamma_\nu(\epsilon)$ that $L_2(\epsilon^{3/2}/(144\gamma_\nu(\epsilon)^{1/2})) \le \sqrt{6}\gamma_\nu(\epsilon)$. Using these, $\sqrt{\epsilon/\gamma_\nu(\epsilon)} \ge 4\|d^k\|$, $\alpha_k = 1$, and (48), we further deduce that

$$\begin{aligned} f(x^{k+1}) - f(x^k) &\overset{(48)}{\le} -\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\|^2 + \frac{L_2(\epsilon^{3/2}/(144\gamma_\nu(\epsilon)^{1/2}))}{12}\sqrt{\frac{\epsilon}{\gamma_\nu(\epsilon)}}\|d^k\|^2 + \frac{\epsilon^{3/2}}{144\gamma_\nu(\epsilon)^{1/2}} \\ &\le -\left(1 - \frac{\sqrt{6}}{12}\right)\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\|^2 + \frac{\epsilon^{3/2}}{144\gamma_\nu(\epsilon)^{1/2}} \le -\frac{\sqrt{\gamma_\nu(\epsilon)\epsilon}}{2}\|d^k\|^2 + \frac{\epsilon^{3/2}}{144\gamma_\nu(\epsilon)^{1/2}} \le -\frac{\epsilon^{3/2}}{144\gamma_\nu(\epsilon)^{1/2}}, \end{aligned}$$

where the second inequality is due to $L_2(\epsilon^{3/2}/(144\gamma_\nu(\epsilon)^{1/2})) \le \sqrt{6}\gamma_\nu(\epsilon)$, and the last inequality follows from $6\|d^k\| \ge \sqrt{\epsilon/\gamma_\nu(\epsilon)}$. Hence, (47) holds as desired.

Case 2) $\alpha_k < 1$. It follows from the definition of α_k in (12) that $4\|d^k\| \ge \sqrt{\epsilon/\gamma_\nu(\epsilon)}$. Recall from (38) with $c_2 = 64$ and $\gamma = \gamma_\nu(\epsilon)$ that $L_2(\epsilon^{3/2}/(64\gamma_\nu(\epsilon)^{1/2})) \le 2\sqrt{6}\gamma_\nu(\epsilon)/3$. By these, the definition of α_k in (12), and (48), we deduce that

$$\begin{aligned} f(x^{k+1}) - f(x^k) &\overset{(48)}{\le} -\alpha_k\sqrt{\gamma_\nu(\epsilon)\epsilon}\|d^k\|^2 + \frac{L_2(\epsilon^{3/2}/(64\gamma_\nu(\epsilon)^{1/2}))}{3}\alpha_k^3\|d^k\|^3 + \frac{\epsilon^{3/2}}{64\gamma_\nu(\epsilon)^{1/2}} \\ &= -\frac{\epsilon^{3/4}\gamma_\nu(\epsilon)^{1/4}}{2}\|d^k\|^{3/2} + \frac{L_2(\epsilon^{3/2}/(64\gamma_\nu(\epsilon)^{1/2}))}{24}\left(\frac{\epsilon}{\gamma_\nu(\epsilon)}\right)^{3/4}\|d^k\|^{3/2} + \frac{\epsilon^{3/2}}{64\gamma_\nu(\epsilon)^{1/2}} \\ &\le -\left(\frac{1}{2} - \frac{\sqrt{6}}{36}\right)\epsilon^{3/4}\gamma_\nu(\epsilon)^{1/4}\|d^k\|^{3/2} + \frac{\epsilon^{3/2}}{64\gamma_\nu(\epsilon)^{1/2}} \le -\frac{\epsilon^{3/4}\gamma_\nu(\epsilon)^{1/4}}{4}\|d^k\|^{3/2} + \frac{\epsilon^{3/2}}{64\gamma_\nu(\epsilon)^{1/2}} \le -\frac{\epsilon^{3/2}}{64\gamma_\nu(\epsilon)^{1/2}}, \end{aligned}$$

where the first equality is due to the definition of α_k, the second inequality is due to $L_2(\epsilon^{3/2}/(64\gamma_\nu(\epsilon)^{1/2})) \le 2\sqrt{6}\gamma_\nu(\epsilon)/3$, and the last inequality follows from $4\|d^k\| \ge \sqrt{\epsilon/\gamma_\nu(\epsilon)}$. Hence, (47) holds as desired. □

The following lemma shows that when the search direction d^k in Algorithm 1 is of type 'NC', the next iterate x^{k+1} produces a sufficient decrease in f.

Lemma 5. *Suppose that Assumption 1 holds and the direction d^k results from the output d of Algorithm 4 with d_type=NC at some iteration k of Algorithm 1. Then $x^{k+1} = x^k + \alpha_k d^k$ satisfies*

$$f(x^k) - f(x^{k+1}) \geq \epsilon^{3/2}/(12\gamma_\nu(\epsilon)^{1/2}). \tag{49}$$

Proof. Since d_type=NC, we see from Lemma 3(ii) that

$$\nabla f(x^k)^T d^k \leq 0, \quad (d^k)^T \nabla^2 f(x^k) d^k / \|d^k\|^2 = -\|d^k\| \leq -\sqrt{\gamma_\nu(\epsilon)\epsilon}. \tag{50}$$

Recall from Theorem 1 that $\{f(x^k)\}_{k\in\mathbb{K}_1}$ is nonincreasing. Thus, $f(x^k + \alpha_k d^k) = f(x^{k+1}) \leq f(x^k)$. Using this, we see that (34) holds for $y = x^k + \alpha_k d^k$ and $x = x^k$. Also, notice from (38) with $c_2 = 12$ and $\gamma = \gamma_\nu(\epsilon)$ that $L_2(\epsilon^{3/2}/(12\gamma_\nu(\epsilon)^{1/2})) \leq \sqrt{2}\gamma_\nu(\epsilon)/2$. Combining these with (50) and $\alpha_k = 1/\gamma_\nu(\epsilon)$, we deduce that

$$\begin{aligned}
f(x^{k+1}) - f(x^k) &\overset{(34)}{\leq} \alpha_k \nabla f(x^k)^T d^k + \frac{\alpha_k^2}{2}(d^k)^T \nabla^2 f(x^k) d^k + \frac{L_2(\epsilon^{3/2}/(12\gamma_\nu(\epsilon)^{1/2}))}{3}\alpha_k^3 \|d^k\|^3 + \frac{\epsilon^{3/2}}{12\gamma_\nu(\epsilon)^{1/2}} \\
&\overset{(50)}{\leq} -\frac{\alpha_k^2}{2}\|d^k\|^3 + \frac{L_2(\epsilon^{3/2}/(12\gamma_\nu(\epsilon)^{1/2}))}{3}\alpha_k^3\|d^k\|^3 + \frac{\epsilon^{3/2}}{12\gamma_\nu(\epsilon)^{1/2}} \\
&= -\frac{1}{2\gamma_\nu(\epsilon)^2}\|d^k\|^3 + \frac{L_2(\epsilon^{3/2}/(12\gamma_\nu(\epsilon)^{1/2}))}{3\gamma_\nu(\epsilon)^3}\|d^k\|^3 + \frac{\epsilon^{3/2}}{12\gamma_\nu(\epsilon)^{1/2}} \\
&\leq -\left(\frac{1}{2} - \frac{\sqrt{2}}{6}\right)\frac{1}{\gamma_\nu(\epsilon)^2}\|d^k\|^3 + \frac{\epsilon^{3/2}}{12\gamma_\nu(\epsilon)^{1/2}} \leq -\frac{1}{6\gamma_\nu(\epsilon)^2}\|d^k\|^3 + \frac{\epsilon^{3/2}}{12\gamma_\nu(\epsilon)^{1/2}} \leq -\frac{\epsilon^{3/2}}{12\gamma_\nu(\epsilon)^{1/2}},
\end{aligned}$$

where the first equality is due to $\alpha_k = 1/\gamma_\nu(\epsilon)$, the third inequality follows from $L_2(\epsilon^{3/2}/(12\gamma_\nu(\epsilon)^{1/2})) \leq \sqrt{2}\gamma_\nu(\epsilon)/2$, and the last inequality follows from $\|d^k\| \geq \sqrt{\gamma_\nu(\epsilon)\epsilon}$. Hence, (49) holds as desired. □

We are now ready to prove Theorem 2.

Proof of Theorem 2. Recall from Theorem 1 that f is nonincreasing along the iterates $\{x^k\}_{k\in\mathbb{K}_1}$ generated by Algorithm 1, which immediately implies that $x^k \in \{x : f(x) \leq f(x^0)\}$ for all $k \in \mathbb{K}_1$. Using this and (4), we have that $\|\nabla^2 f(x^k)\| \leq U_H$ for all $k \in \mathbb{K}_1$.

(i) Suppose for contradiction that the total number of iterations of Algorithm 1 is more than K_1. Observe from Algorithm 1 and Lemmas 4 and 5 that each iteration except the last one results in a reduction on the function value of f at least by $\epsilon^{3/2}/(144\gamma_\nu(\epsilon)^{1/2})$. Hence,

$$K_1\epsilon^{3/2}/(144\gamma_\nu(\epsilon)^{1/2}) \leq \sum_{k\in\mathbb{K}_1}[f(x^k) - f(x^{k+1})] \leq f(x^0) - f_{\text{low}},$$

where $\mathbb{K}_1$ is given in Theorem 1. This leads to a contradiction with the definition of K_1 in (13).

(ii) By Theorem 7 with $(H, \varepsilon) = (\nabla^2 f(x^k), \sqrt{\gamma_\nu(\epsilon)\epsilon})$ and the fact that $\|\nabla^2 f(x^k)\| \leq U_H$, we can observe that the number of gradient evaluations and Hessian-vector products of f required by each call of Algorithm 4 in Algorithm 1 is at most $\widetilde{\mathcal{O}}(\min\{n, [\gamma_\nu(\epsilon)\epsilon]^{-1/4}\})$. Also, notice that each iteration of Algorithm 1 requires one call of Algorithm 4. Combining these with statement (i), we see that statement (ii) holds. □

7.2 Proof of the main results in Section 4

In this subsection, we provide a proof of Theorems 3 and 4.

The following lemma gives some useful properties of the output of Algorithm 4, which is identical to Lemma 3 with $(x^k, d^k, \gamma_\nu(\epsilon))$ replaced by $(\tilde{x}, \tilde{d}^t, \tilde{\gamma}_t)$.

Lemma 6. *Suppose that Assumption 1 holds and the direction $\tilde{d}^t$ results from the output d of Algorithm 4 with a type specified in d_type at some iteration of Algorithm 2. Then the following statements hold.*

(i) *If d_type=SOL, then* $\tilde{d}^t$ *satisfies*

$$\sqrt{\tilde{\gamma}_t\epsilon}\|\tilde{d}^t\|^2 \le (\tilde{d}^t)^T(\nabla^2 f(\tilde{x}) + 2\sqrt{\tilde{\gamma}_t\epsilon}I)\tilde{d}^t, \tag{51}$$

$$(\tilde{d}^t)^T\nabla f(\tilde{x}) = -(\tilde{d}^t)^T(\nabla^2 f(\tilde{x}) + 2\sqrt{\tilde{\gamma}_t\epsilon}I)\tilde{d}^t, \tag{52}$$

$$\|(\nabla^2 f(\tilde{x}) + 2\sqrt{\tilde{\gamma}_t\epsilon}I)\tilde{d}^t + \nabla f(\tilde{x})\| \le \zeta\sqrt{\tilde{\gamma}_t\epsilon}\|\tilde{d}^t\|/2. \tag{53}$$

(ii) *If d_type=NC, then* $\tilde{d}^t$ *satisfies* $\nabla f(\tilde{x})^T\tilde{d}^t \le 0$ *and*

$$(\tilde{d}^t)^T\nabla^2 f(\tilde{x})\tilde{d}^t/\|\tilde{d}^t\|^2 = -\|\tilde{d}^t\| \le -\sqrt{\tilde{\gamma}_t\epsilon}. \tag{54}$$

We now provide a proof of Theorem 3.

Proof of Theorem 3. Notice that if Algorithm 2 breaks the inner loop at $t = 0$, the conclusion of this theorem clearly holds. We now suppose for the rest of the proof that Algorithm 2 does not break its inner loop at $t = 0$. Claim that for all $t \ge 0$ that Algorithm 2 does not break its inner loop, it holds that $\tilde{\gamma}_t \le \gamma_\nu(\epsilon)$. Indeed, suppose that Algorithm 2 does not break its inner loop for some $t \ge 0$, and that $\tilde{d}^t$ along with d_type is generated from the output of Algorithm 4. Recall from Lemma 6 that if d_type=SOL, then (51)-(53) hold for $\tilde{d}^t$, and we see from $\|\nabla f(x^k)\| > \epsilon$, $\tilde{x} = x^k$, and (53) that $\tilde{d}^t \ne 0$. If d_type=NC, we see from Lemma 6(ii) that $\tilde{d}^t \ne 0$, and moreover,

$$\nabla f(\tilde{x})^T\tilde{d}^t \le 0, \quad (\tilde{d}^t)^T\nabla^2 f(\tilde{x})\tilde{d}^t/\|\tilde{d}^t\|^2 = -\|\tilde{d}^t\| \le -\sqrt{\tilde{\gamma}_t\epsilon}. \tag{55}$$

We next show that $\tilde{\gamma}_t \le \gamma_\nu(\epsilon)$ holds by considering five separate cases below.

Case 1) d_type=SOL and $f(\tilde{x} + \tilde{\alpha}_t\tilde{d}^t) > f(\tilde{x})$. Since d_type=SOL, we see that (51)-(53) hold for $\tilde{d}^t$. Using similar arguments as for (43) with $(x^k, d^k, \gamma_\nu(\epsilon))$ replaced by $(\tilde{x}, \tilde{d}^t, \tilde{\gamma}_t)$, we have that $\tilde{\gamma}_t \le H_\nu^{2/(1+\nu)}\epsilon^{-(1-\nu)/(1+\nu)}$. Combining this with the definition of $\gamma_\nu(\epsilon)$ in (9), we obtain that $\tilde{\gamma}_t \le \gamma_\nu(\epsilon)$.

Case 2) d_type=SOL, $f(\tilde{x} + \tilde{\alpha}_t\tilde{d}^t) \le f(\tilde{x})$, and $\tilde{\alpha}_t = 1$. It follows from d_type=SOL that (51)-(53) hold for $\tilde{d}^t$. Using $\tilde{\alpha}_t = 1$ and the definition of $\tilde{\alpha}_t$ in (17), we have $4\|\tilde{d}^t\| \le \sqrt{\epsilon/\tilde{\gamma}_t}$. In addition, it follows from $f(\tilde{x} + \tilde{\alpha}_t\tilde{d}^t) \le f(\tilde{x})$ and $\tilde{\alpha}_t = 1$ that (33) holds for $y = \tilde{x} + \tilde{d}^t$ and $x = \tilde{x}$. Since Algorithm 2 does not break its inner loop with $\tilde{\alpha}_t = 1$ and $\tilde{d}^t$, we see that $\|\nabla f(\tilde{x} + \tilde{d}^t)\| > \epsilon$. In view of these and (53), we see that

$$\begin{aligned}\epsilon < \|\nabla f(\tilde{x} + \tilde{d}^t)\| &\le \|\nabla f(\tilde{x} + \tilde{d}^t) - \nabla f(\tilde{x}) - \nabla^2 f(\tilde{x})\tilde{d}^t\| \\ &\quad + \|(\nabla^2 f(\tilde{x}) + 2\sqrt{\tilde{\gamma}_t\epsilon}I)\tilde{d}^t + \nabla f(\tilde{x})\| + 2\sqrt{\tilde{\gamma}_t\epsilon}\|\tilde{d}^t\| \\ &\overset{(33)(53)}{\le} \frac{L_1(\epsilon/4)}{2}\|\tilde{d}^t\|^2 + \frac{\epsilon}{4} + \frac{\zeta+4}{2}\sqrt{\tilde{\gamma}_t\epsilon}\|\tilde{d}^t\| \\ &\le \frac{L_1(\epsilon/4)\epsilon}{32\tilde{\gamma}_t} + \frac{\epsilon}{4} + \frac{(\zeta+4)\epsilon}{8} \le \frac{L_1(\epsilon/4)\epsilon}{32\tilde{\gamma}_t} + \frac{7\epsilon}{8},\end{aligned}$$

where the fourth inequality is due to $4\|\tilde{d}^t\| \le \sqrt{\epsilon/\tilde{\gamma}_t}$, and the last inequality follows from $\zeta \in (0,1)$. Rearranging the terms of this inequality and using (37) with $c_1 = 4$, we derive that

$$\tilde{\gamma}_t \le L_1(\epsilon/4)/4 \le \gamma_\nu(\epsilon)/8 < \gamma_\nu(\epsilon),$$

where the second inequality is due to (37).

Case 3) d_type=SOL, $f(\tilde{x} + \tilde{\alpha}_t\tilde{d}^t) \le f(\tilde{x})$, and $\tilde{\alpha}_t < 1$. Since d_type=SOL, we see that (51)-(53) hold for $\tilde{d}^t$. Using $\tilde{\alpha}_t < 1$ and the definition of $\tilde{\alpha}_t$ in (17), we have $4\|\tilde{d}^t\| > \sqrt{\epsilon/\tilde{\gamma}_t}$. In addition, it follows from $f(\tilde{x} + \tilde{\alpha}_t\tilde{d}^t) \le f(\tilde{x})$ that (34) holds for $y = \tilde{x} + \tilde{\alpha}_t\tilde{d}^t$ and $x = \tilde{x}$. By the same arguments as for (48) with $(x^{k+1}, x^k, \alpha_k, d^k, \gamma_\nu(\epsilon))$ replaced by $(\tilde{x} + \tilde{\alpha}_t\tilde{d}^t, \tilde{x}, \tilde{\alpha}_t, \tilde{d}^t, \tilde{\gamma}_t)$, we can see that for any $\delta > 0$,

$$f(\tilde{x} + \tilde{\alpha}_t\tilde{d}^t) - f(\tilde{x}) \le -\tilde{\alpha}_t\sqrt{\tilde{\gamma}_t\epsilon}\|\tilde{d}^t\|^2 + \frac{L_2(\delta)}{3}\tilde{\alpha}_t^3\|\tilde{d}^t\|^3 + \delta, \tag{56}$$

Since Algorithm 2 does not break its inner loop with $\tilde{d}^t$ and $\tilde{\alpha}_t$, we see that $\tilde{d}^t$ and $\tilde{\alpha}_t$ violate (19). In view of this, $\tilde{\alpha}_t < 1$, and the definition of $\tilde{\alpha}_t$ in (17), $4\|\tilde{d}^t\| > \sqrt{\epsilon/\tilde{\gamma}_t}$, and (56) with $\delta = \epsilon^{3/2}/(64\tilde{\gamma}_t^{1/2})$, we see that

$$\begin{aligned}\frac{\epsilon^{3/4}\tilde{\gamma}_t^{1/4}\|\tilde{d}^t\|^{3/2}}{4} = \frac{\sqrt{\tilde{\gamma}_t\epsilon}\,\tilde{\alpha}_t\|\tilde{d}^t\|^2}{2} &\le f(\tilde{x}+\tilde{\alpha}_t\tilde{d}^t) - f(\tilde{x}) + \tilde{\alpha}_t\sqrt{\tilde{\gamma}_t\epsilon}\|\tilde{d}^t\|^2 \overset{(56)}{\le} \frac{L_2(\epsilon^{3/2}/(64\tilde{\gamma}_t^{1/2}))}{3}\tilde{\alpha}_t^3\|\tilde{d}^t\|^3 + \frac{\epsilon^{3/2}}{64\tilde{\gamma}_t^{1/2}} \\ &= \frac{L_2(\epsilon^{3/2}/(64\tilde{\gamma}_t^{1/2}))}{24}\left(\frac{\epsilon}{\tilde{\gamma}_t}\right)^{3/4}\|\tilde{d}^t\|^{3/2} + \frac{\epsilon^{3/2}}{64\tilde{\gamma}_t^{1/2}} \\ &\le \frac{L_2(\epsilon^{3/2}/(64\tilde{\gamma}_t^{1/2}))}{24}\left(\frac{\epsilon}{\tilde{\gamma}_t}\right)^{3/4}\|\tilde{d}^t\|^{3/2} + \frac{\epsilon^{3/4}\tilde{\gamma}_t^{1/4}\|\tilde{d}^t\|^{3/2}}{8},\end{aligned}$$

where the first and second equalities are due to the definition of $\tilde{\alpha}_t$ in (17) and $\tilde{\alpha}_t < 1$, the first inequality is due to the violation of (19), and the last inequality follows from $4\|\tilde{d}^t\| > \sqrt{\epsilon/\tilde{\gamma}_t}$. Rearranging the terms of this inequality and using the fact that $\|\tilde{d}^t\| \neq 0$, we obtain that

$$L_2(\epsilon^{3/2}/(64\tilde{\gamma}_t^{1/2})) \ge 3\tilde{\gamma}_t > 2\sqrt{6}\tilde{\gamma}_t/3.$$

In view of this and (38) with $c_2 = 64$, we see that $\tilde{\gamma}_t < \gamma_\nu(\epsilon)$.

Case 4) d_type=NC and $f(\tilde{x}+\tilde{\alpha}_t\tilde{d}^t) > f(\tilde{x})$. It follows from d_type=NC that (55) holds. Using similar arguments as for (46) with $(x^k, d^k, \gamma_\nu(\epsilon))$ replaced by $(\tilde{x}, \tilde{d}^t, \tilde{\gamma}_t)$, we obtain that $\tilde{\gamma}_t \le H_\nu^{2/(1+\nu)}\epsilon^{-(1-\nu)/(1+\nu)}$. Combining this with the definition of $\gamma_\nu(\epsilon)$ in (9), we obtain that $\tilde{\gamma}_t \le \gamma_\nu(\epsilon)$.

Case 5) d_type=NC and $f(\tilde{x}+\tilde{\alpha}_t\tilde{d}^t) \le f(\tilde{x})$. It follows from d_type=NC that (55) holds. In addition, since Algorithm 2 does not break its inner loop with $\tilde{\alpha}_t$ and $\tilde{d}^t$, we see that $\tilde{\alpha}_t$ and $\tilde{d}^t$ violate (16). By $\tilde{\alpha}_t = 1/\tilde{\gamma}_t$, $f(\tilde{x}+\tilde{\alpha}_t\tilde{d}^t) \le f(\tilde{x})$, one sees that (34) holds for $y = \tilde{x}+\tilde{\alpha}_t\tilde{d}^t$ and $x = \tilde{x}$. In view of these, we have

$$\begin{aligned}-\frac{\tilde{\alpha}_t^2\|\tilde{d}^t\|^3}{6} &< f(\tilde{x}+\tilde{\alpha}_t\tilde{d}^t) - f(\tilde{x}) \overset{(34)}{\le} \tilde{\alpha}_t\nabla f(\tilde{x})^T\tilde{d}^t + \frac{\tilde{\alpha}_t^2}{2}(\tilde{d}^t)^T\nabla^2 f(\tilde{x})\tilde{d}^t + \frac{L_2(\epsilon^{3/2}/(6\tilde{\gamma}_t^{1/2}))}{3}\tilde{\alpha}_t^3\|\tilde{d}^t\|^3 + \frac{\epsilon^{3/2}}{6\tilde{\gamma}_t^{1/2}} \\ &\overset{(55)}{\le} -\frac{\tilde{\alpha}_t^2}{2}\|\tilde{d}^t\|^3 + \frac{L_2(\epsilon^{3/2}/(6\tilde{\gamma}_t^{1/2}))}{3}\tilde{\alpha}_t^3\|\tilde{d}^t\|^3 + \frac{\epsilon^{3/2}}{6\tilde{\gamma}_t^{1/2}} \\ &= -\frac{1}{2\tilde{\gamma}_t^2}\|\tilde{d}^t\|^3 + \frac{L_2(\epsilon^{3/2}/(6\tilde{\gamma}_t^{1/2}))}{3\tilde{\gamma}_t^3}\|\tilde{d}^t\|^3 + \frac{\epsilon^{3/2}}{6\tilde{\gamma}_t^{1/2}} \le -\frac{1}{3\tilde{\gamma}_t^2}\|\tilde{d}^t\|^3 + \frac{L_2(\epsilon^{3/2}/(6\tilde{\gamma}_t^{1/2}))}{3\tilde{\gamma}_t^3}\|\tilde{d}^t\|^3,\end{aligned}$$

where the first inequality is due the the violation of (16), the first equality is due to $\tilde{\alpha}_t = 1/\tilde{\gamma}_t$, and the last inequality follows from $\|\tilde{d}^t\| \ge \sqrt{\tilde{\gamma}_t\epsilon}$ (see (55)). In view of this inequality and the fact that $\tilde{d}^t \neq 0$, we see that

$$L_2(\epsilon^{3/2}/(6\tilde{\gamma}_t^{1/2})) > \tilde{\gamma}_t/2.$$

In view of this and (38) with $c_2 = 6$, we see that $\tilde{\gamma}_t < \gamma_\nu(\epsilon)$.

Combining the above five cases, we obtain that $\tilde{\gamma}_t \le \gamma_\nu(\epsilon)$ holds if Algorithm 2 does not break its inner loop with $\tilde{\alpha}_t$ and $\tilde{d}^t$. By this and $\tilde{\gamma}_t = \theta^t\tilde{\gamma} \ge \theta^t\gamma_{-1}$, we see that the number of calls of Algorithm 4 at iteration k of Algorithm 2 is bounded above by T. Suppose that $\gamma_k = \tilde{\gamma}_{T_k}$ for some $1 \le T_k \le T$. We see that Algorithm 2 does not break its inner loop at $t = T_k - 1$, which implies $\gamma_k = \theta\tilde{\gamma}_{T_k-1} \le \theta\gamma_\nu(\epsilon)$. Hence, the conclusion of this theorem holds as desired. □

The next lemma shows that when the search direction d^k in Algorithm 2 is of type 'SOL', the next iterate x^{k+1} produces a sufficient decrease in f.

Lemma 7. *Suppose that Assumption 1 holds and the direction d^k results from the output d of Algorithm 4 with d_type=SOL at some iteration k of Algorithm 2. Then $x^{k+1} = x^k + \alpha_k d^k$ satisfies either $\|\nabla f(x^{k+1})\| \le \epsilon$ or*

$$f(x^k) - f(x^{k+1}) \ge \epsilon^{3/2}/(72\gamma_k^{1/2}). \qquad (57)$$

Proof. Since d_type=SOL, one can see from Algorithm 2 and Lemma 6(i) that (51)-(53) hold with $(\tilde{x}, \tilde{d}^t, \tilde{\gamma}_t)$ replaced by (x^k, d^k, γ_k). Notice that if $\|\nabla f(x^{k+1})\| \leq \epsilon$, the conclusion of this lemma holds. Hence, it suffices to show that (57) holds if $\|\nabla f(x^{k+1})\| > \epsilon$. To this end, we suppose for the rest of the proof that $\|\nabla f(x^{k+1})\| > \epsilon$, and consider two separate cases below.

Case 1) $\alpha_k = 1$. By this, $(d^k, \alpha_k, \gamma_k) = (\tilde{d}^t, \tilde{\alpha}_t, \tilde{\gamma}_t)$, and the definition of $\tilde{\alpha}_t$ in (17), one has $4\|d^k\| \leq \sqrt{\epsilon/\gamma_k}$. Since $\alpha_k = 1$ and $\|\nabla f(x^{k+1})\| = \|\nabla f(x^k + d^k)\| > \epsilon$, we observe from Algorithm 2 that

$$f(x^{k+1}) \leq f(x^k) - \sqrt{\gamma_k \epsilon}\|d^k\|^2/2, \tag{58}$$

$$\|\nabla f(x^k + d^k) - \nabla f(x^k) - \nabla^2 f(x^k) d^k\| \leq 2\gamma_k \|d^k\|^2 + \epsilon/2. \tag{59}$$

In view of these, $(\tilde{x}, \tilde{d}^t, \tilde{\gamma}_t) = (x^k, d^k, \gamma_k)$, and (53), we see that

$$\begin{aligned}\epsilon < \|\nabla f(x^k + d^k)\| &\leq \|\nabla f(x^k + d^k) - \nabla f(x^k) - \nabla^2 f(x^k) d^k\| \\ &\quad + \|(\nabla^2 f(x^k) + 2\sqrt{\gamma_k \epsilon} I) d^k + \nabla f(x^k)\| + 2\sqrt{\gamma_k \epsilon}\|d^k\| \\ &\overset{(53)(59)}{\leq} 2\gamma_k \|d^k\|^2 + \frac{\epsilon}{2} + \frac{4+\zeta}{2}\sqrt{\gamma_k \epsilon}\|d^k\| \leq \frac{5+\zeta}{2}\sqrt{\gamma_k \epsilon}\|d^k\| + \frac{\epsilon}{2},\end{aligned}$$

where the last inequality follows from $4\|d^k\| \leq \sqrt{\epsilon/\gamma_k}$. It together with $\zeta \in (0,1)$ follows that $6\|d^k\| \geq \sqrt{\epsilon/\gamma_k}$, which along with (58) implies that $f(x^k) - f(x^{k+1}) \geq \epsilon^{3/2}/(72\gamma_k^{1/2})$. Hence, the inequality (57) holds as desired.

Case 2) $\alpha_k < 1$. By this, $(d^k, \alpha_k, \gamma_k) = (\tilde{d}^t, \tilde{\alpha}_t, \tilde{\gamma}_t)$, and the definition of $\tilde{\alpha}_t$ in (17), one has $4\|d^k\| > \sqrt{\epsilon/\gamma_k}$. Since $\alpha_k < 1$, we see from Algorithm 2 that $f(x^{k+1}) \leq f(x^k) - \sqrt{\gamma_k \epsilon}\alpha_k^2 \|d^k\|^2/2$. In view of these, and the definition of α_k in (17), we see that

$$f(x^k) - f(x^{k+1}) \geq \sqrt{\gamma_k \epsilon}\alpha_k^2 \|d^k\|^2/2 = \epsilon\|d^k\|/8 > \epsilon^{3/2}/(32\gamma_k^{1/2}),$$

where the first equality is due to the definition of α_k, and the last inequality follows from $4\|d^k\| > \sqrt{\epsilon/\gamma_k}$. Hence, the inequality (57) holds as desired.

□

Our next lemma shows that when the search direction d^k in Algorithm 2 is of type 'NC', the next iterate x^{k+1} produces a sufficient decrease in f.

Lemma 8. *Suppose that Assumption 1 holds and the direction d^k results from the output d of Algorithm 4 with d_type=NC at some iteration k of Algorithm 2. Then $x^{k+1} = x^k + \alpha_k d^k$ satisfies*

$$f(x^k) - f(x^{k+1}) \geq \epsilon^{3/2}/(6\gamma_k^{1/2}). \tag{60}$$

Proof. Since d_type=NC, we see from Algorithm 2 and Lemma 6(ii) that

$$(d^k)^T \nabla^2 f(x^k) d^k / \|d^k\|^2 = -\|d^k\| \leq -\sqrt{\gamma_k \epsilon}. \tag{61}$$

In addition, notice from Algorithm 2 that $f(x^{k+1}) \leq f(x^k) - \alpha_k^2 \|d^k\|^3/6$. Using this, $\|d^k\| \geq \sqrt{\gamma_k \epsilon}$ (see (61)), and $\alpha_k = 1/\gamma_k$, we obtain that

$$f(x^k) - f(x^{k+1}) \geq \alpha_k^2 \|d^k\|^3/6 = \|d^k\|^3/(6\gamma_k^2) \geq \epsilon^{3/2}/(6\gamma_k^{1/2}),$$

where the last inequality is due to $\|d^k\| \geq \sqrt{\gamma_k \epsilon}$. Hence, (60) holds as desired. □

We are now ready to prove Theorem 4.

Proof of Theorem 4. For notational convenience, we let $\{x^k\}_{k \in \mathbb{K}_2}$ denote all the iterates generated by Algorithm 2, where $\mathbb{K}_2$ is a set of consecutive nonnegative integers starting from 0. Notice that f is descent along the iterates generated by Algorithm 2, which implies that $x^k \in \{x : f(x) \leq f(x^0)\}$ for all $k \in \mathbb{K}_2$. It then follows from (4) that $\|\nabla^2 f(x^k)\| \leq U_H$ for all $k \in \mathbb{K}_2$.

(i) Suppose for contradiction that the total number of iterations of Algorithm 2 is more than K_2. Recall from Theorem 3 that $\gamma_k \leq \bar{\gamma}_\nu(\epsilon)$ holds for all $k \in \mathbb{K}_2$. It then follows from Algorithm 2 and Lemmas 7 and 8 that each iteration except the last one results in a reduction on the function value of f at least by $\epsilon^{3/2}/(72\bar{\gamma}_\nu(\epsilon)^{1/2})$. Hence,

$$K_2\epsilon^{3/2}/(72\bar{\gamma}_\nu(\epsilon)^{1/2}) \le \sum_{k\in\mathbb{K}_2}[f(x^k)-f(x^{k+1})] \le f(x^0)-f_{\mathrm{low}},$$

which contradicts (22). Therefore, the total number of iterations of Algorithm 2 is at most K_2.

(ii) From statement (i), we see that Algorithm 2 terminates at some iteration $\underline{K}$ satisfying $\underline{K} \le K_2$. It follows from Algorithm 2 and Lemmas 7 and 8 that the kth iteration with $k < \underline{K}$ of Algorithm 2 results in a reduction on the function value of f at least by $\epsilon^{3/2}/(72\gamma_k^{1/2})$. Hence, $\sum_{k=0}^{\underline{K}-2}\epsilon^{3/2}/(72\gamma_k^{1/2}) \le \sum_{k=0}^{\underline{K}-2}[f(x^k)-f(x^{k+1})] \le f(x^0)-f_{\mathrm{low}}$, which then implies that $\sum_{k=0}^{\underline{K}-2}1/\gamma_k^{1/2} \le 72(f(x^0)-f_{\mathrm{low}})\epsilon^{-3/2}$. Using this and the Cauchy-Schwarz inequality, we deduce that

$$\left(\sum_{k=0}^{\underline{K}-2}1/\gamma_k^{1/4}\right)^2 \le \left(\sum_{k=0}^{\underline{K}-2}1/\gamma_k^{1/2}\right)(\underline{K}-1) \le 72(f(x^0)-f_{\mathrm{low}})\epsilon^{-3/2}(\underline{K}-1). \tag{62}$$

On the other hand, notice that $\tilde{\gamma}_t \ge \tilde{\gamma}_0 = \max\{\gamma_{-1}, \gamma_{k-1}/\theta\}$ for all $\tilde{\gamma}_t$ generated at iteration k of Algorithm 2. In view of this, $\|\nabla^2 f(x^k)\| \le U_H$, and Theorem 3, we can see that the number of gradient evaluations and Hessian-vector products of f required by one call of Algorithm 4 with $(H,\varepsilon,g) = (\nabla^2 f(x^k), (\tilde{\gamma}_t\epsilon)^{1/2}, \nabla f(x^k))$ at iteration k of Algorithm 2 is bounded above by

$$\min\left\{n, \left\lceil\left(\sqrt{\frac{U_H}{(\tilde{\gamma}_t\epsilon)^{1/2}}}+2\right)\psi\left(\frac{U_H}{(\tilde{\gamma}_t\epsilon)^{1/2}}\right)\right\rceil\right\} \le \min\left\{n, \left(\sqrt{\frac{U_H}{(\gamma_{k-1}\epsilon/\theta)^{1/2}}}+2\right)\psi\left(\frac{U_H}{(\gamma_{-1}\epsilon)^{1/2}}\right)+1\right\}.$$

where the inequality follows from $\tilde{\gamma}_t \ge \max\{\gamma_{-1}, \gamma_{k-1}/\theta\}$ and the monotonicity of ψ. Recall from Theorem 3, the number of calls of Algorithm 4 at iteration k of Algorithm 2 is at most T. Combining these, we obtain that the total number of gradient evaluations and Hessian-vector products of f required by Algorithm 2 is bounded by

$$\begin{aligned}
&\sum_{k=0}^{\underline{K}-1} T\min\left\{n, \left(\sqrt{\frac{U_H}{(\gamma_{k-1}\epsilon/\theta)^{1/2}}}+2\right)\psi\left(\frac{U_H}{(\gamma_{-1}\epsilon)^{1/2}}\right)+1\right\}\\
&\le T\min\left\{n\underline{K}, \sum_{k=0}^{\underline{K}-1}\left[\left(\sqrt{\frac{U_H}{(\gamma_{k-1}\epsilon/\theta)^{1/2}}}+2\right)\psi\left(\frac{U_H}{(\gamma_{-1}\epsilon)^{1/2}}\right)+1\right]\right\}\\
&= T\min\left\{n\underline{K}, \psi\left(\frac{U_H}{(\gamma_{-1}\epsilon)^{1/2}}\right)\frac{U_H^{1/2}}{(\epsilon/\theta)^{1/4}}\sum_{k=0}^{\underline{K}-1}\frac{1}{\gamma_{k-1}^{1/4}} + \left[2\psi\left(\frac{U_H}{(\gamma_{-1}\epsilon)^{1/2}}\right)+1\right]\underline{K}\right\}\\
&= \widetilde{\mathcal{O}}\left(T\min\left\{n\underline{K}, \epsilon^{-1/4}\left(\sum_{k=0}^{\underline{K}-1}1/\gamma_{k-1}^{1/4}\right)+\underline{K}\right\}\right) = \widetilde{\mathcal{O}}(T\min\{n\underline{K}, \epsilon^{-1}\underline{K}^{1/2}+\underline{K}\}),
\end{aligned} \tag{63}$$

where the first inequality is due to $\min\{a_1,a_2\}+\min\{b_1,b_2\} \le \min\{a_1+a_2, b_1+b_2\}$ for all $a_1,a_2,b_1,b_2 \in \mathbb{R}$, the second equality follows from the definition of ψ in Theorem 7, and the last equality is due to

$$\sum_{k=0}^{\underline{K}-1}1/\gamma_{k-1}^{1/4} = 1/\gamma_{-1}^{1/4} + \sum_{k=0}^{\underline{K}-2}1/\gamma_k^{1/4} \overset{(62)}{\le} 1/\gamma_{-1}^{1/4} + \sqrt{72f(x^0-f_{\mathrm{low}})\epsilon^{-3/2}(\underline{K}-1)}.$$

It then follows from (63) and $\underline{K} \le K_2$ that the conclusion of the statement (ii) holds.

□

7.3 Proof of the main results in Section 5

In this subsection, we provide a proof of Theorems 5 and 6.

We first provide a proof of Theorem 5.

Proof of Theorem 5. We prove this theorem by induction. Let x^k, x^{k+1} be two consecutive iterates generated by Algorithm 3, and suppose that $\{f(x^\ell)\}_{0\le\ell\le k}$ is nonincreasing. We next prove $f(x^{k+1}) \le f(x^k)$. By similar arguments as those used in Theorem 1, we see that $f(x^{k+1}) \le f(x^k)$ when x^{k+1} is generated by α_k and d^k resulting from the outputs of Algorithm 4. Thus, it remains to show that $f(x^{k+1}) \le f(x^k)$ when x^{k+1} is generated by α_k and d^k resulting from the outputs of Algorithm 5. Suppose for contradiction that $f(x^{k+1}) = f(x^k+\alpha_k d^k) > f(x^k)$ in this case. Let $\varphi(\alpha) = f(x^k + \alpha d^k)$. Then $\varphi(\alpha_k) > \varphi(0)$. Since d^k results from the output v of Algorithm 5, we see from Algorithm 3 that

$$\nabla f(x^k)^T d^k \le 0, \quad (d^k)^T\nabla^2 f(x^k)d^k/\|d^k\|^2 = -\|d^k\| \le -\epsilon_H/2 < 0. \tag{64}$$

By this and the definition of φ, we see that $\varphi'(0) = \nabla f(x^k)^T d^k \le 0$ and $\varphi''(0) = (d^k)^T\nabla^2 f(x^k)d^k < 0$. Since $\varphi(\alpha_k) > \varphi(0)$, it then follows that there exists a local minimizer $\alpha_* \in (0, \alpha_k)$ of φ such that $\varphi(\alpha_*) < \varphi(0)$. By the second order optimality condition of φ at α_*, we have $\varphi''(\alpha_*) = (d^k)^T\nabla^2 f(x^k+\alpha_* d^k)d^k \ge 0$. In addition, by $f(x^k) \le f(x^0)$ and $\varphi(\alpha_*) < \varphi(0)$, one has $f(x^k + \alpha_* d^k) < f(x^0)$. Hence, (4) holds for $x = x^k$ and $y = x^k + \alpha_* d^k$. By these, we obtain that

$$\begin{aligned} H_\nu\alpha_*^\nu\|d^k\|^{2+\nu} &\ge \|d^k\|^2\|\nabla^2 f(x^k+\alpha_* d^k) - \nabla^2 f(x^k)\| \ge (d^k)^T(\nabla^2 f(x^k+\alpha_* d^k) - \nabla^2 f(x^k))d^k \\ &\ge -(d^k)^T\nabla^2 f(x^k)d^k = \|d^k\|^3. \end{aligned}$$

Recall from (64) that $\|d^k\| \ge \epsilon_H/2$. It then follows from the above inequality, $d^k \ne 0$ and $\alpha_k > \alpha_*$ that $\alpha_k > \alpha_* \ge H_\nu^{-1/\nu}(\epsilon_H/2)^{(1-\nu)/\nu}$, which contradicts the definition of α_k in (26). □

The following lemma shows that when the search direction d^k in Algorithm 3 is a negative curvature direction returns from Algorithm 5, the next iterate x^{k+1} produces a sufficient reduction in f.

Lemma 9. *Suppose that Assumption 1 holds with $\nu \in (0,1]$ and the direction d^k results from the output v of Algorithm 5 at some iteration k of Algorithm 3. Then $x^{k+1} = x^k + \alpha_k d^k$ satisfies*

$$f(x^k) - f(x^{k+1}) \ge \frac{(\epsilon_H/2)^{(2+\nu)/\nu}}{4(2H_\nu)^{2/\nu}}. \tag{65}$$

Proof. Since d^k results from the output v of Algorithm 5, we see from Algorithm 3 that (64). Notice from Theorem 5 that $f(x^k + \alpha_k d^k) \le f(x^k) \le f(x^0)$. Hence, (6) holds for $x = x^k$ and $y = x^k + \alpha_k d^k$. By this and (64), one has

$$\begin{aligned} f(x^{k+1}) - f(x^k) &\overset{(6)}{\le} \alpha_k\nabla f(x^k)^T d^k + \frac{\alpha_k^2}{2}(d^k)^T\nabla^2 f(x^k)d^k + \frac{H_\nu\alpha_k^{2+\nu}\|d^k\|^{2+\nu}}{(1+\nu)(2+\nu)} \\ &\overset{(64)}{\le} -\frac{\alpha_k^2}{2}\|d^k\|^3 + \frac{H_\nu}{2}\alpha_k^{2+\nu}\|d^k\|^{2+\nu} = -\frac{\alpha_k^2}{2}\|d^k\|^3 + \frac{\alpha_k^2}{4}\left(\frac{\epsilon_H}{2}\right)^{1-\nu}\|d^k\|^{2+\nu} \\ &\le -\frac{\alpha_k^2}{4}\|d^k\|^3 \le -\frac{(\epsilon_H/2)^{(2+\nu)/\nu}}{4(2H_\nu)^{2/\nu}}, \end{aligned}$$

where the third inequality is due to $\|d^k\| \ge \epsilon_H/2$, and the last inequality follows from $\|d^k\| \ge \epsilon_H/2$ and the definition of α_k in (26). Hence, the inequality (65) holds as desired. □

We are now ready to provide a proof of Theorem 6.

Proof of Theorem 6. Recall from Theorem 5 that f is nonincreasing along the iterates $\{x^k\}_{k\in\mathbb{K}_3}$ generated by Algorithm 3. Thus, $x^k \in \{x : f(x) \le f(x^0)\}$. It then follows from (4) that $\|\nabla^2 f(x^k)\| \le U_H$ for all $k \in \mathbb{K}_3$. In addition, observe that Algorithm 3 proceeds in the same manner as Algorithm 1 when $\|\nabla f(x^k)\| > \epsilon_g$ at some iteration k. Therefore, at such iteration k, we have that Lemmas 4 and 5 hold with ϵ replaced by ϵ_g.

(i) Suppose for contradiction that the total number of calls of Algorithm 5 in Algorithm 3 is more than $\widetilde{K}_2$. Notice from Algorithm 3 and Lemma 9 that each of these calls except the last one, returns a sufficiently negative curvature direction, and each of them results in a reduction on f of at least $(\epsilon_H/2)^{(2+\nu)/\nu}/[4(2H_\nu)^{2/\nu}]$. Hence,

$$\widetilde{K}_2(\epsilon_H/2)^{(2+\nu)/\nu}/[4(2H_\nu)^{2/\nu}] \le \sum_{k\in\mathbb{K}_3}[f(x^k) - f(x^{k+1})] \le f(x^0) - f_{\text{low}},$$

which contradicts the definition of $\widetilde{K}_2$ given in (28). Hence, statement (i) holds.

(ii) Suppose for contradiction that the total number of calls of Algorithm 4 in Algorithm 3 is more than $\widetilde{K}_1$. Observe that if Algorithm 4 is called at some iteration k and generates the next iterate x^{k+1} satisfying $\|\nabla f(x^{k+1})\| \le \epsilon_g$, then Algorithm 5 must be called at the next iteration $k+1$. In view of this and statement (i), we see that the total number of such iterations k is at most $\widetilde{K}_2$. Hence, the total number of iterations k of Algorithm 3 at which Algorithm 4 is called and generates the next iterate x^{k+1} satisfying $\|\nabla f(x^{k+1})\| > \epsilon_g$ is at least $\widetilde{K}_1 - \widetilde{K}_2 + 1$. Moreover, for each of such iterations k, we observe from Lemmas 4 and 5 with ϵ replaced by ϵ_g that $f(x^k) - f(x^{k+1}) \ge \epsilon_g^{3/2}/[144\gamma_\nu(\epsilon_g)]$. It then follows that

$$(\widetilde{K}_1 - \widetilde{K}_2 + 1)\epsilon_g^{3/2}/[144\gamma_\nu(\epsilon_g)] \le \sum_{k\in\mathbb{K}_3} [f(x^k) - f(x^{k+1})] \le f(x^0) - f_{\mathrm{low}},$$

which contradicts the definition of $\widetilde{K}_1$ and $\widetilde{K}_2$ given in (27) and (28), respectively.

(iii) Notice that either Algorithm 4 or Algorithm 5 is called at each iteration of Algorithm 3. It follows from this and statements (i) and (ii) that the total number of iterations of Algorithm 3 is at most $\widetilde{K}_1 + \widetilde{K}_2$. In addition, one can also easily observe that the output x^k of Algorithm 3 satisfies $\|\nabla f(x^k)\| \le \epsilon_g$ deterministically and $\lambda_{\min}(\nabla^2 f(x^k)) \ge -\epsilon_H$ with probability at least $1-\delta$ for some $0 \le k \le \widetilde{K}_1 + \widetilde{K}_2$, where the latter part is due to Algorithm 5. Hence, statement (iii) holds as desired.

(iv) By Theorem 7 with $(H,\varepsilon) = (\nabla^2 f(x^k), \sqrt{\gamma_\nu(\epsilon_g)\epsilon_g})$ and the fact that $\|\nabla^2 f(x^k)\| \le U_H$, we observe that the number of gradient evaluations and Hessian-vector products of f required by each call of Algorithm 4 with input $U = 0$ is at most $\widetilde{\mathcal{O}}(\min\{n, [\gamma_\nu(\epsilon_g)\epsilon_g]^{-1/4}\})$. In addition, by Theorem 8 with $(H,\varepsilon) = (\nabla^2 f(x^k), \epsilon_H)$, $\|\nabla^2 f(x^k)\| \le U_H$, and the fact that each iteration of the Lanczos method requires only one matrix-vector product, one can observe that the number of Hessian-vector products of f required by each call of Algorithm 5 is also at most $\widetilde{\mathcal{O}}(\min\{n, \epsilon_H^{-1/2}\})$. Based on these and statement (iii), we see that statement (iv) holds. □

8 Future work

There are several possible extensions of this work. First, it would be interesting to study the iteration and operation complexity of second-order methods for nonconvex constrained optimization with Hölder continuous Hessian. Second, more numerical studies would be helpful to further improve the proposed Newton-CG methods from a practical perspective. Lastly, the development of a parameter-free method that achieves the best-known iteration and operation complexity bounds for finding an approximate SOSP of problem (1) remains an open question.

References

[1] N. Agarwal, Z. Allen-Zhu, B. Bullins, E. Hazan, and T. Ma. Finding approximate local minima faster than gradient descent. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1195–1199, 2017.

[2] Z. Allen-Zhu and Y. Li. Neon2: Finding local minima via first-order oracles. *Advances in Neural Information Processing Systems*, 31, 2018.

[3] A. E. Beaton and J. W. Tukey. The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics*, 16(2):147–185, 1974.

[4] E. G. Birgin and J. M. Martínez. The use of quadratic regularization with a cubic descent condition for unconstrained optimization. *SIAM J. Optim.*, 27(2):1049–1074, 2017.

[5] R. H. Byrd, F. E. Curtis, and J. Nocedal. Infeasibility detection and SQP methods for nonlinear optimization. *SIAM J. Optim.*, 20(5):2281–2299, 2010.

[6] Y. Carmon and J. Duchi. Gradient descent finds the cubic-regularized nonconvex Newton step. *SIAM J. Optim.*, 29(3):2146–2178, 2019.

[7] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. “Convex until proven guilty”: Dimension-free acceleration of gradient descent on non-convex functions. In *International conference on machine learning*, pages 654–663. PMLR, 2017.

[8] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Accelerated methods for nonconvex optimization. *SIAM J. Optim.*, 28(2):1751–1772, 2018.

[9] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points I. *Math. Program.*, 184(1-2):71–120, 2020.

[10] C. Cartis, N. I. Gould, and P. L. Toint. Adaptive cubic regularisation methods for unconstrained optimization. part i: motivation, convergence and numerical results. *Math. Program.*, 127(2):245–295, 2011.

[11] C. Cartis, N. I. Gould, and P. L. Toint. Worst-case evaluation complexity and optimality of second-order methods for nonconvex smooth optimization. In *Proceedings of the International Congress of Mathematicians: Rio de Janeiro 2018*, pages 3711–3750. World Scientific, 2018.

[12] F. E. Curtis, D. P. Robinson, C. W. Royer, and S. J. Wright. Trust-region Newton-CG with strong second-order complexity guarantees for nonconvex optimization. *SIAM J. Optim.*, 31(1):518–544, 2021.

[13] F. E. Curtis, D. P. Robinson, and M. Samadi. A trust region algorithm with a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$ for nonconvex optimization. *Math. Program.*, 162:1–32, 2017.

[14] F. E. Curtis, D. P. Robinson, and M. Samadi. An inexact regularized Newton framework with a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$ for nonconvex optimization. *IMA J. Numer. Anal.*, 39(3):1296–1327, 2019.

[15] P. Dvurechensky. Gradient method with inexact oracle for composite non-convex optimization. *arXiv preprint arXiv:1703.09180*, 2017.

[16] G. N. Grapiglia and Y. Nesterov. Regularized Newton methods for minimizing functions with Hölder continuous Hessians. *SIAM J. Optim.*, 27(1):478–506, 2017.

[17] C. He, Z. Lu, and T. K. Pong. A newton-cg based augmented lagrangian method for finding a second-order stationary point of nonconvex equality constrained optimization with complexity guarantees. *arXiv preprint arXiv:2301.03139*, 2023.

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

[19] C. Jin, P. Netrapalli, and M. I. Jordan. Accelerated gradient descent escapes saddle points faster than gradient descent. In *Conference On Learning Theory*, pages 1042–1085. PMLR, 2018.

[20] J. Kuczyński and H. Woźniakowski. Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start. *SIAM J. Matrix Anal. Appl.*, 13(4):1094–1122, 1992.

[21] B. Li, S. Tang, and H. Yu. Better approximations of high dimensional smooth functions by deep neural networks with rectified power units. *arXiv preprint arXiv:1903.05858*, 2019.

[22] H. Li and Z. Lin. Restarted nonconvex accelerated gradient descent: No more polylogarithmic factor in the $\mathcal{O}(\epsilon^{-7/4})$ complexity. In *International Conference on Machine Learning*, pages 12901–12916. PMLR, 2022.

[23] J. M. Martínez and M. Raydan. Cubic-regularization counterpart of a variable-norm trust-region method for unconstrained minimization. *J. Glob. Optim.*, 68:367–385, 2017.

[24] N. Marumo and A. Takeda. Parameter-free accelerated gradient descent for nonconvex minimization. *arXiv preprint arXiv:2212.06410*, 2022.

[25] Y. Nesterov. Universal gradient methods for convex optimization problems. *Math. Program.*, 152(1-2):381–404, 2015.

[26] Y. Nesterov and B. T. Polyak. Cubic regularization of Newton method and its global performance. *Math. Program.*, 108(1):177–205, 2006.

[27] M. O'Neill and S. J. Wright. A log-barrier Newton-CG method for bound constrained optimization with complexity guarantees. *IMA J. Numer. Anal.*, 41(1):84–121, 2021.

[28] C. W. Royer, M. O'Neill, and S. J. Wright. A Newton-CG algorithm with complexity guarantees for smooth unconstrained optimization. *Math. Program.*, 180(1-2):451–488, 2020.

[29] C. W. Royer and S. J. Wright. Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization. *SIAM J. Optim.*, 28(2):1448–1477, 2018.

[30] Y. Xu, R. Jin, and T. Yang. Neon+: Accelerated gradient methods for extracting negative curvature for non-convex optimization. *arXiv preprint arXiv:1712.01033*, 2017.

[31] C. Zhang and R. Jiang. Riemannian adaptive regularized Newton methods with Hölder continuous Hessians. *arXiv preprint arXiv:2309.04052*, 2023.

Appendix

A A capped conjugate gradient method

In this part we present the capped CG method proposed in [28, Algorithm 1] for finding either an approximate solution to the linear system (7) or a sufficiently negative curvature direction of the associated matrix H, which has been briefly discussed in Section 3. Its details can be found in [28, Section 3.1].

The following theorem presents the iteration complexity of Algorithm 4.

Theorem 7 (iteration complexity of Algorithm 4). *Consider applying Algorithm 4 with input $U = 0$ to the linear system (7) with $g \neq 0$, $\varepsilon > 0$, and H being an $n \times n$ symmetric matrix. Then the number of iterations of Algorithm 4 is at most*

$$\min\left\{n, \left\lceil \left(\sqrt{\|H\|/\varepsilon} + 2\right) \psi\left(\|H\|/\varepsilon\right)\right\rceil \right\} = \widetilde{\mathcal{O}}(\min\{n, \sqrt{\|H\|/\varepsilon}\}),$$

where $\psi(t) = \ln(144(\sqrt{t+2}+1)^2(t+2)^6/\zeta^2)$.

Proof. From [28, Lemma 1], we know that the number of iterations of Algorithm 4 is bounded by $\min\{n, J(U,\varepsilon,\zeta)\}$, where $J(U,\varepsilon,\zeta)$ is the smallest integer J such that $\sqrt{T}\tau^{J/2} \le \widehat{\zeta}$, with $U, \widehat{\zeta}, T$ and τ being the values returned by Algorithm 4. In addition, it was shown in [28, Section 3.1] that $J(U,\varepsilon,\zeta) \le \lceil (\sqrt{\kappa} + 1/2) \ln\left(144(\sqrt{\kappa}+1)^2\kappa^6/\zeta^2\right)\rceil$, where $\kappa = U/\varepsilon + 2$ is an output by Algorithm 4. Also, observe that $\sqrt{\kappa} \le \sqrt{U/\varepsilon} + \sqrt{2} \le \sqrt{U/\varepsilon} + 3/2$. Combining these, we obtain that $J(U,\varepsilon,\zeta) \le \left\lceil \left(\sqrt{U/\varepsilon} + 2\right) \ln\left(144(\sqrt{U/\varepsilon+2}+1)^2(U/\varepsilon+2)^6/\zeta^2\right)\right\rceil$. Notice from Algorithm 4 that the output $U \le \|H\|$. Using these, we obtain that the conclusion holds as desired. □

B A randomized Lanczos based minimum eigenvalue oracle

In this part we present the randomized Lanczos method proposed in [28, Section 3.2], which can be used as a minimum eigenvalue oracle for Algorithm 2. As briefly discussed in Section 5, this oracle outputs either a sufficiently negative curvature direction of H or a certificate that H is nearly positive semidefinite with high probability. More detailed motivation and explanation of it can be found in [28, Section 3.2].

The following theorem justifies that Algorithm 5 is a suitable minimum eigenvalue oracle for Algorithm 2. Its proof is identical to that of [28, Lemma 2] and thus omitted.

Theorem 8 (iteration complexity of Algorithm 5). *Consider Algorithm 5 with tolerance $\varepsilon > 0$, probability parameter $\delta \in (0,1)$, and symmetric matrix $H \in \mathbb{R}^{n\times n}$ as its input. Then it either finds a sufficiently negative curvature direction v satisfying $v^T H v \le -\varepsilon/2$ and $\|v\| = 1$ or certifies that $\lambda_{\min}(H) \ge -\varepsilon$ holds with probability at least $1-\delta$ in at most $N(\varepsilon,\delta)$ iterations, where $N(\varepsilon,\delta)$ is defined in (66).*

Algorithm 4 A capped conjugate gradient method

```
*Inputs*: symmetric matrix $H \in \mathbb{R}^{n \times n}$, vector $g \neq 0$, damping parameter $\varepsilon > 0$, desired relative accuracy $\zeta \in (0,1)$.
*Optional input:* scalar $U \geq 0$ (set to 0 if not provided).
*Outputs:* d_type, $d$.
*Secondary outputs:* final values of $U, \kappa, \widehat{\zeta}, \tau$, and $T$.
Set
    $$\bar{H} := H + 2\varepsilon I, \quad \kappa := \frac{U + 2\varepsilon}{\varepsilon}, \quad \widehat{\zeta} := \frac{\zeta}{3\kappa}, \quad \tau := \frac{\sqrt{\kappa}}{\sqrt{\kappa}+1}, \quad T := \frac{4\kappa^4}{(1-\sqrt{\tau})^2},$$
$y^0 \leftarrow 0, r^0 \leftarrow g, p^0 \leftarrow -g, j \leftarrow 0.$
**if** $(p^0)^T \bar{H} p^0 < \varepsilon \|p^0\|^2$ **then**
    Set $d \leftarrow p^0$ and terminate with d_type = NC;
**else if** $\|Hp^0\| > U\|p^0\|$ **then**
    Set $U \leftarrow \|Hp^0\|/\|p^0\|$ and update $\kappa, \widehat{\zeta}, \tau, T$ accordingly;
**end if**
**while** TRUE **do**
    $\alpha_j \leftarrow (r^j)^T r^j / (p^j)^T \bar{H} p^j$; {Begin Standard CG Operations}
    $y^{j+1} \leftarrow y^j + \alpha_j p^j$;
    $r^{j+1} \leftarrow r^j + \alpha_j \bar{H} p^j$;
    $\beta_{j+1} \leftarrow \|r^{j+1}\|^2 / \|r^j\|^2$;
    $p^{j+1} \leftarrow -r^{j+1} + \beta_{j+1} p^j$; {End Standard CG Operations}
    $j \leftarrow j + 1$;
    **if** $\|Hp^j\| > U\|p^j\|$ **then**
        Set $U \leftarrow \|Hp^j\|/\|p^j\|$ and update $\kappa, \widehat{\zeta}, \tau, T$ accordingly;
    **end if**
    **if** $\|Hy^j\| > U\|y^j\|$ **then**
        Set $U \leftarrow \|Hy^j\|/\|y^j\|$ and update $\kappa, \widehat{\zeta}, \tau, T$ accordingly;
    **end if**
    **if** $\|Hr^j\| > U\|r^j\|$ **then**
        Set $U \leftarrow \|Hr^j\|/\|r^j\|$ and update $\kappa, \widehat{\zeta}, \tau, T$ accordingly;
    **end if**
    **if** $(y^j)^T \bar{H} y^j < \varepsilon \|y^j\|^2$ **then**
        Set $d \leftarrow y^j$ and terminate with d_type = NC;
    **else if** $\|r^j\| \leq \widehat{\zeta} \|r^0\|$ **then**
        Set $d \leftarrow y^j$ and terminate with d_type = SOL;
    **else if** $(p^j)^T \bar{H} p^j < \varepsilon \|p^j\|^2$ **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 $\alpha_j, y^{j+1}$ as in the main loop above;
        Find $i \in \{0, \dots, j-1\}$ such that
            $$(y^{j+1} - y^i)^T \bar{H} (y^{j+1} - y^i) < \varepsilon \|y^{j+1} - y^i\|^2;$$
        Set $d \leftarrow y^{j+1} - y^i$ and terminate with d_type = NC;
    **end if**
**end while**
```

Notice that $\|H\|$ is required in Algorithm 5. In general, computing $\|H\|$ may not be cheap when n is large. Nevertheless, $\|H\|$ can be efficiently estimated via a randomization scheme with high confidence (e.g., see the discussion in [28, Appendix B3]).

Algorithm 5 A randomized Lanczos based minimum eigenvalue oracle

Input: symmetric matrix $H \in \mathbb{R}^{n\times n}$, tolerance $\varepsilon > 0$, and probability parameter $\delta \in (0, 1)$.

Output: a sufficiently negative curvature direction v satisfying $v^T Hv \leq -\varepsilon/2$ and $\|v\| = 1$; or a certificate that $\lambda_{\min}(H) \geq -\varepsilon$ with probability at least $1 - \delta$.

Apply the Lanczos method [20] to estimate $\lambda_{\min}(H)$ starting with a random vector uniformly generated on the unit sphere, and run it for at most

$$N(\varepsilon, \delta) := \min\left\{n, 1 + \left\lceil \frac{\ln(2.75n/\delta^2)}{2}\sqrt{\frac{\|H\|}{\varepsilon}}\right\rceil\right\} \tag{66}$$

iterations. If a unit vector v with $v^T Hv \leq -\varepsilon/2$ is found at some iteration, terminate immediately and return v.