Journal of the Operations Research Society of China
https://doi.org/10.1007/s40305-022-00398-5

Cubic Regularization Methods with Second-Order Complexity Guarantee Based on a New Subproblem Reformulation

Ru-Jun Jiang[1] · Zhi-Shuo Zhou[1] · Zi-Rui Zhou[2]

Received: 26 June 2021 / Revised: 16 December 2021 / Accepted: 13 February 2022

Abstract

The cubic regularization (CR) algorithm has attracted a lot of attentions in the literature in recent years. We propose a new reformulation of the cubic regularization subproblem. The reformulation is an unconstrained convex problem that requires computing the minimum eigenvalue of the Hessian. Then, based on this reformulation, we derive a variant of the (non-adaptive) CR provided a known Lipschitz constant for the Hessian and a variant of adaptive regularization with cubics (ARC). We show that the iteration complexity of our variants matches the best-known bounds for unconstrained minimization algorithms using first- and second-order information. Moreover, we show that the operation complexity of both of our variants also matches the state-of-the-art bounds in the literature. Numerical experiments on test problems from CUTEst collection show that the ARC based on our new subproblem reformulation is comparable to the existing algorithms.

This paper is dedicated to the late Professor Duan Li in commemoration of his contributions to optimization, financial engineering, and risk management.

The first author is supported in part by the National Natural Foundation of China (Nos. 11801087 and 12171100).

✉ Ru-Jun Jiang
rjjiang@fudan.edu.cn

Zhi-Shuo Zhou
zhouzs18@fudan.edu.cn

Zi-Rui Zhou
zirui.zhou@huawei.com

1 School of Data Science, Fudan University, Shanghai 200433, China

2 Huawei Technologies Canada, Burnaby, BC, Canada

Published online: 15 April 2022

Keywords Cubic regularization subproblem · First-order methods · Constrained convex optimization · Complexity analysis

Mathematics Subject Classification 65K05 · 90C26 · 90C30

1 Introduction

Consider the generic unconstrained optimization problem

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

where $f : \mathbb{R}^n \to \mathbb{R}$ is a twice Lipschitz continuously differentiable and possibly nonconvex function. Recently, the cubic regularization (CR) algorithm [1, 2] or its variants has attracted a lot of attentions for solving problem (1), due to its practical efficiency and elegant theoretical convergence guarantees. Each iteration of the CR solves the following subproblem

$$\min_{s \in \mathbb{R}^n} m(s) := \frac{1}{2} s^\top H s + g^\top s + \frac{\sigma}{3} \|s\|^3 , \tag{CRS}$$

where H and g represent the Hessian and gradient of the function f at the current iterate, respectively, $\|\cdot\|$ denotes the Euclidean l_2 norm, H is an $n \times n$ symmetric matrix (possibly non-positive semidefinite) and σ is a regularization parameter that may be adaptive during the iterations. This model can be seen as a second-order Taylor expansion plus a cubic regularizer that makes the next iterate not too far away from the current iterate. It is well known that under mild conditions ([1, 2]), the CR converges to a point satisfying the second-order necessary condition (SONC), i.e.,

$$\nabla f(x) = 0, \quad \nabla^2 f(x) \succeq 0,$$

where $(\cdot) \succeq 0$ means $(\cdot)$ is a positive semidefinite matrix. In the literature, it is of great interests to find a weaker condition than SONC, i.e.,

$$\|\nabla f(x)\| < \epsilon_g, \quad \lambda_{\min}(\nabla^2 f(x)) \geqslant -\epsilon_H, \quad \epsilon_g, \epsilon_H > 0, \tag{2}$$

where $\lambda_{\min}(H)$ denotes the minimum eigenvalue for a matrix H. Condition (2) is often said to be (ϵ_g, ϵ_H) stationary.

The CR algorithm was first considered by Griewank in an unpublished technical report ([3]). Nesterov and Polyak [1] proposed the CR in a different perspective and demonstrated that it takes $\mathcal{O}(\epsilon_g^{-3/2})$ iterations to find an $(\epsilon_g, \epsilon_g^{1/2})$ stationary point if each subproblem is solved exactly. As in general the Lipschitz constant of the Hessian is difficult to estimate, Cartis et al. [2, 4] proposed an adaptive version of the CR algorithm, called the ARC (adaptive regularization with cubics), and showed that it admits an iteration complexity bound $\mathcal{O}\left(\max\{\epsilon_g^{-3/2}, \epsilon_H^{-3}\}\right)$ to find an (ϵ_g, ϵ_H)

stationary point, when the subproblems are solved inexactly and the regularization parameter $\sigma > 0$ is chosen adaptively.

Besides iteration complexity $\mathcal{O}(\epsilon_g^{-3/2})$, many subsequent studies proposed variants of the CR or other second-order methods that also have an *operation complexity* $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ (where $\tilde{\mathcal{O}}(\cdot)$ hides the logarithm factors), with high probability, for finding an $(\epsilon_g, \epsilon_g^{1/2})$ stationary point of problem (1). Here, a unit operation can be a function evaluation, gradient evaluation, Hessian evaluation or a matrix vector product ([5]). Based on the CR algorithm, Agarwal et al. [6] derived an algorithm with such an operation complexity bound, where the heart of the algorithm is a subproblem solver that returns, with high probability, an approximate solution to the problem (CRS) in $\tilde{\mathcal{O}}(\epsilon_g^{-1/4})$ operations. After that, Carmon et al. [7] proposed an accelerated gradient method that also converges to an $(\epsilon_g, \epsilon_g^{1/2})$ stationary point with an operation complexity $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$. Royer and Wright [8] proposed a hybrid algorithm that combines Newton-like steps, the CG method for inexactly solving linear systems, and the Lanczos procedure for approximately computing negative curvature directions, which was shown to have an operation complexity $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ to achieve an $(\epsilon_g, \epsilon_g^{1/2})$ stationary point. Royer et al. [9] proposed a variant of Newton-CG algorithm with the same complexity guarantee. Very recently, Curtis et al. [5] considered a variant of trust-region Newton methods based on inexactly solving the trust-region subproblem by the well-known "trust-region Newton-conjugate gradient" method, whose complexity also matches the-state-of-the-art. All the above-mentioned methods [5, 7–9] converge with high probability like [6], which is due to the use of randomized iterative methods for approximately computing the minimum eigenvalue, e.g., the Lanczos procedure.

Despite theoretical guarantees, the practical efficiency of solving (CRS) heavily affects the convergence of the CR algorithm. Although it is one of the most successful algorithms for solving (CRS) in practice, the Krylov subspace method ([2]) may fail to converge to the true solution of (CRS) in the hard case[1] or close to being in the hard case. Carmon and Duchi [10] provided the first convergence rate analysis of the Krylov subspace method in the easy case, based on which the authors further propose a CR algorithm with an operation complexity $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ in [11]. Carmon and Duchi [12] also showed the gradient descent method that works in both the easy and hard cases is able to converge to the global minimizer if the step size is sufficiently small, though the convergence rate is worse than the Krylov subspace method. Based on a novel convex reformulation of (CRS), Jiang et al. [13] proposed an accelerated first-order algorithm that works efficiently in practice in both the easy and hard cases, and meanwhile enjoys theoretical guarantees of the same order with the Krylov subspace method.

However, the methods in the literature ([1, 2, 4, 6–9, 11, 13]), either somehow deviate the framework of the CR or ARC algorithms, and/or do not present good practical performance and an $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ operation complexity simultaneously. Our goal in this paper is to propose variants of the CR and ARC based on new subproblem reformulations that achieve the state-of-the-art complexity bounds and also remain close

[1] For the problem (CRS), it is said to be in the easy if the optimal solution x^* satisfies $\rho\|x^*\| > -\lambda_{\min}(A)$, and hard case otherwise.

to the practically efficient CR and ARC algorithms. Motivated by the reformulation in [13], we deduce a new unconstrained convex reformulation for (CRS). Our reformulation explores hidden convexity of (CRS), where similar ideas also appear in the (generalized) trust-region subproblem ([14–17]). The main cost of the reformulation is computing the minimum eigenvalue of the Hessian. We propose a variant of the CR algorithm with strong complexity guarantee. We consider the more realistic case where eigenvalues of Hessians are computed inexactly. In this setting, we suppose the Lipschitz constant of the Hessian is given as L, the parameter $\sigma = L/2$ is non-adaptive, and each subproblem is also solved approximately. We prove that our algorithm converges to an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point with an iteration complexity $\mathcal{O}(\epsilon_g^{-3/2})$. Moreover, we further show that each iteration costs $\tilde{\mathcal{O}}(\epsilon_g^{-1/4})$ when the minimum eigenvalue of the Hessian is inexactly computed by the Lanczos procedure, and the subproblem, which is regularized to be strongly convex, is approximately solved by Nesterov's accelerated gradient method (NAG) [18] in each iteration. Combining the above facts, we further demonstrate that our algorithm has an *operation complexity* $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ for finding an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point. Based on the reformulation, we also propose a variant of the ARC with similar iteration and operation complexity guarantees, where σ_k is adaptive in each iteration.

The remaining of this paper is organized as follows. In Sect. 2, we derive our unconstrained convex reformulation for (CRS), describe the CR and ARC algorithms and the basic setting, and give unified convergence analysis for sufficient decrease in the model function in one iteration. In Sects. 3 and 4, we give convergence analysis for the CR and ARC algorithms for finding an approximate second-order stationary point with both iteration complexity and operation complexity bounds that match the best-known ones, respectively. In Sect. 5, we compare numerical performance of an ARC embedded by our reformulation with ARCs based on the existing subproblem solvers. We conclude our paper in Sect. 6.

2 Preliminaries

The structure of this section is as follows. In Sect. 2.1, we first propose our reformulation for the subproblem (CRS). Then, in Sect. 2.2, we mainly describe the framework of our variants of the CR and ARC algorithms and also state our convergence results. Finally, in Sect. 2.3, we give unified convergence analysis of one iteration progress for both the CR and ARC algorithms.

2.1 A New Convex Reformulation for (CRS)

In this subsection, we introduce a new reformulation for (CRS) when $\lambda_{\min}(H) < 0$, i.e., the minimum eigenvalue of H is negative. First, recall the reformulation proposed in [13]:

$$\min_{s,y} g^\top s + \frac{1}{2}s^\top (H - \alpha I) s + \frac{\sigma}{3} y^{3/2} + \frac{\alpha}{2} y \\ \text{s.t. } y \geqslant \|s\|^2, \quad y \geqslant \frac{\alpha^2}{\sigma^2}, \tag{3}$$

where $\alpha = \lambda_{\min}(H)$. However, this reformulation may be ill-conditioned and cause numerical instability when y is small since the Hessian of the objective function for y is $\frac{\sigma}{4}y^{-1/2}$, which approaches infinity when $y \to 0$. Unfortunately, this is the case for the CR or ARC algorithms when the iteration number k becomes large. We also found that due to this issue and that y is of the same order with $\|s\|^2$, CR or ARC based on solving subproblem (3) cannot achieve the state-of-the-art operation complexity $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ for finding an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point. To amend this issue, we proposed the following reformulation,

$$\min_{s,y} \hat{m}(s, y) := g^\top s + \frac{1}{2}s^\top (H - \alpha I) s + \frac{\sigma}{3} y^3 + \frac{\alpha}{2} y^2 \\ \text{s.t. } y \geqslant \|s\|, \quad y \geqslant -\frac{\alpha}{\sigma}, \tag{CRS$_r$}$$

so that y is of the same order with $\|s\|$.

One key observation of this paper is that (CRS$_r$) can be simplified into a convex problem with single variable s, by applying partial minimization on y. Note that given any $s \in \mathbb{R}^n$, the y-problem of (CRS$_r$) is

$$\min_{y\in\mathbb{R}} \left\{ \frac{\sigma}{3}y^3 + \frac{\alpha}{2}y^2 : \ y \geqslant \|s\|, \ y \geqslant -\frac{\alpha}{\sigma} \right\},$$

whose optimal solution is uniquely given by

$$y = \max\left\{\|s\|, -\frac{\alpha}{\sigma}\right\}. \tag{4}$$

This is because the derivative of the objective function is $\sigma y^2 + \alpha y$, satisfying

$$\sigma y^2 + \alpha y = \sigma y \left(y + \frac{\alpha}{\sigma}\right) \geqslant 0,$$

due to the constraints $y \geqslant 0$ and $y \geqslant -\frac{\alpha}{\sigma}$. Substituting (4) into (CRS$_r$), we obtain that (CRS$_r$) is equivalent to

$$\min_{s\in\mathbb{R}^n} \left\{ \tilde{m}(s) := g^\top s + \frac{1}{2}s^\top (H - \alpha I)s + J_{\alpha,\sigma}(s) \right\}, \tag{CRS$_u$}$$

where

$$J_{\alpha,\sigma}(s) = \frac{\sigma}{3}\left[\max\left\{\|s\|, -\frac{\alpha}{\sigma}\right\}\right]^3 + \frac{\alpha}{2}\left[\max\left\{\|s\|, -\frac{\alpha}{\sigma}\right\}\right]^2. \tag{5}$$

In the following, we show that $J_{\alpha,\sigma}(s)$ is a convex and continuously differentiable function.

Proposition 1 *For any $\sigma > 0$ and $\alpha \in \mathbb{R}$, $J_{\alpha,\sigma}(s)$ is convex and continuously differentiable on $\mathbb{R}^n$. Moreover, we have*

$$\nabla J_{\alpha,\sigma}(s) = [\sigma \|s\| + \alpha]_+ \cdot s, \qquad \forall s \in \mathbb{R}^n,$$

where $[a]_+ = \max\{a, 0\}$ for any $a \in \mathbb{R}$.

Proof We consider the two cases (a) $\alpha \geqslant 0$ and (b) $\alpha < 0$ separately.

(a) If $\alpha \geqslant 0$, then by $\sigma > 0$, we have $\|s\| \geqslant -\alpha/\sigma$ for all $s \in \mathbb{R}^n$. Thus, $J_{\alpha,\sigma}(s)$ reduces to

$$J_{\alpha,\sigma}(s) = \frac{\sigma}{3}\|s\|^3 + \frac{\alpha}{2}\|s\|^2, \quad \forall s \in \mathbb{R}^n.$$

It is clear that in this case $J_{\alpha,\sigma}(s)$ is convex and continuously differentiable, and

$$\nabla J_{\alpha,\sigma}(s) = \sigma\|s\|s + \alpha s = (\sigma\|s\| + \alpha) \cdot s = [\sigma\|s\| + \alpha]_+ \cdot s,$$

where the last equality is due to $\alpha \geqslant 0$.

(b) Now we consider the case $\alpha < 0$. Note that the following identity holds for any $\sigma > 0, \alpha, y \in \mathbb{R}$:

$$\frac{\sigma}{3}y^3 + \frac{\alpha}{2}y^2 = \frac{\sigma}{3}\left(y + \frac{\alpha}{\sigma}\right)^3 - \frac{\alpha}{2}\left(y + \frac{\alpha}{\sigma}\right)^2 + \frac{\alpha^3}{6\sigma^2}.$$

By this, we can rewrite $J_{\alpha,\sigma}(s)$ in (5) as

$$J_{\alpha,\sigma}(s) = \frac{\sigma}{3}\left[\|s\| + \frac{\alpha}{\sigma}\right]_+^3 - \frac{\alpha}{2}\left[\|s\| + \frac{\alpha}{\sigma}\right]_+^2 + \frac{\alpha^3}{6\sigma^2}. \tag{6}$$

Note that $\|s\| + \alpha/\sigma$ is a convex function of s. In addition, $[\cdot]_+^3$ and $[\cdot]_+^2$ are both non-decreasing convex functions. Thus, we obtain that

$$h_1(s) := \left[\|s\| + \frac{\alpha}{\sigma}\right]_+^3 \quad \text{and} \quad h_2(s) := \left[\|s\| + \frac{\alpha}{\sigma}\right]_+^2$$

are convex functions. This, together with $\alpha < 0$ and (6), implies that $J_{\alpha,\sigma}(s)$ is convex. Also, it is easy to verify that

$$\nabla h_1(s) = \begin{cases} 0, & \text{if } \|s\| \leqslant -\frac{\alpha}{\sigma}; \\ 3\left(\|s\| + \frac{\alpha}{\sigma}\right)^2 \cdot \frac{s}{\|s\|}, & \text{if } \|s\| > -\frac{\alpha}{\sigma}. \end{cases}$$

$$\nabla h_2(s) = \begin{cases} 0, & \text{if } \|s\| \leqslant -\frac{\alpha}{\sigma}; \\ 2\left(\|s\| + \frac{\alpha}{\sigma}\right) \cdot \frac{s}{\|s\|}, & \text{if } \|s\| > -\frac{\alpha}{\sigma}. \end{cases}$$

This, together with (6), implies that

$$\nabla J_{\alpha,\sigma}(s) = \begin{cases} 0, & \text{if } \|s\| \leqslant -\frac{\alpha}{\sigma}; \\ (\sigma\|s\| + \alpha) \cdot s, & \text{if } \|s\| > -\frac{\alpha}{\sigma}. \end{cases}$$

It then follows that $J_{\alpha,\sigma}(s)$ is continuously differentiable and

$$\nabla J_{\alpha,\sigma}(s) = [\sigma\|s\| + \alpha]_+ \cdot s.$$

Combining the results in cases (a) and (b), we complete the proof.

We immediately have the following results.

Corollary 1 *The $\tilde{m}(s)$ in (CRS_u) is convex and continuously differentiable, and*

$$\nabla\tilde{m}(s) = g + (H - \alpha I)s + [\sigma\|s\| + \alpha]_+ s.$$

Moreover, if $\sigma\|s\| + \alpha \geqslant 0$, we have

$$m(s) = \tilde{m}(s) \;\; and \;\; \nabla m(s) = \nabla\tilde{m}(s).$$

2.2 Variants of the CR and the ARC Algorithms and Main Complexity Results

In this subsection, we first summarize our variants of the CR and ARC algorithms in Algorithms 1 and 2. Note that the only difference between Algorithms 1 and 2 is that Algorithm 2 has an adaptive regularizer σ_k in the model function, where the Hessian Lipschitz constant L is replaced by the adaptive parameter $2\sigma_k$, and thus Algorithm 2 needs carefully choosing parameters related to σ_k.

Before presenting the convergence analysis, we give some general assumptions and conditions that are widely used in the literature. We first introduce the following assumption for the objective function, which was used in [19].

Assumption 1 The function f is twice differentiable with $f^* = \min_x f(x)$, and has bounded and Lipschitz continuous Hessian on the piece-wise linear path generated by the iterates, i.e., there exists $L > 0$ such that

$$\|\nabla^2 f(x) - \nabla^2 f(x_k)\| \leqslant L\|x - x_k\|, \quad \forall x \in [x_k, x_k + d_k], \tag{11}$$

where x_k is the kth iterate and d_k is the kth update. Here $\|A\|$ denotes the operator 2-norm for a matrix A.

An immediate result of Assumption 1 is the following well-known cubic upper bound for any $s \in \mathbb{R}^n$ (cf. equation (1.1) in [2])

$$f(x_k + s) - f(x_k) \leqslant g_k^\top s + \frac{1}{2}s^\top H_k s + \frac{L}{6}\|s\|^3. \tag{12}$$

Algorithm 1 A variant of the CR algorithm using reformulation (CRS$_u$)

Require $x_0,\ \epsilon_g > 0,\ L > 0, \epsilon_E = \sqrt{L\epsilon_g}/3$ and $\epsilon_S = \epsilon_g/9$
1 **for** $k = 0, 1, \cdots,$ **do**
2 evaluate $g_k = \nabla f(x_k),\ H_k = \nabla^2 f(x_k)$,
3 compute an approximate eigenpair (α_k, v_k) such that $\alpha_k = v_k^\top H_k v_k \leqslant \lambda_{\min}(H_k) + \epsilon_E$
4 **if** $\|g_k\| \leqslant \epsilon_g$ and $\alpha_k \geqslant -2\epsilon_E$ **then**
5 return x_k
6 **end if**
7 **if** $\alpha_k \geqslant -\epsilon_E$ **then**
8 solve the regularized subproblem approximately

$$s_k \approx \mathrm{argmin}_{s\in\mathbb{R}^n} \left\{ m_k^r(s) := g_k^\top s + \frac{1}{2} s^\top (H_k + 3\epsilon_E I)s + \frac{L}{6}\|s\|^3 \right\}, \tag{7}$$

9 $d_k = s_k, x_{k+1} = x_k + d_k$
10 **else**
11 solve the regularized subproblem approximately

$$s_k \approx \mathrm{argmin}_{s\in\mathbb{R}^n} \left\{ \tilde{m}_k^r(s) := g_k^\top s + \frac{1}{2} s^\top (H_k - \alpha_k I + 2\epsilon_E I)s + \tilde{J}_k(s) \right\}, \tag{8}$$

where $\tilde{J}_k(s) = J_{\alpha_k, L/2}(s)$
12 **if** $L\|s_k\| + 2\alpha_k \geqslant 0$ **then**
13 $d_k = s_k$
14 **else**
15 $w_k = \beta v_k$ such that $\|w_k\| = |\alpha_k|$ and $w_k^\top g_k \leqslant 0$
16 $d_k = \frac{1}{L} w_k$
17 **end if**
18 $x_{k+1} = x_k + d_k$
19 **end if**
20 **end for**

As in practice, it is expensive to compute the exact smallest eigenvalue, we consider the case that the smallest eigenvalue is approximately computed. Note that in line 3 of Algorithm 1 (and line 4 of Algorithm 2), we call an approximate eigenvalue solver to find an approximate eigenvalue α_k and a unit vector v_k such that

$$\lambda_{\min}(H_k) \leqslant \alpha_k = v_k^\top H_k v_k \leqslant \lambda_{\min}(H_k) + \epsilon_E.$$

To make the model function ϵ_E-strongly convex, we add $\frac{3}{2}\epsilon_E\|s_k\|^2$ to m_k or $\epsilon_E\|s_k\|^2$ to $\tilde{m}_k$ (denoted by m_k^r or $\tilde{m}_k^r$), i.e.,

$$m_k^r(s) := g_k^\top s + \frac{1}{2} s^\top (H_k + 3\epsilon_E I)s + \frac{\sigma}{3}\|s\|^3$$

and

$$\tilde{m}_k^r(s) := g_k^\top s + \frac{1}{2} s^\top (H_k - \alpha_k I + 2\epsilon_E I)s + \tilde{J}_k(s),$$

Algorithm 2 A variant of the ACR algorithm using reformulation (CRS$_u$)

Require $x_0,\ 2 > \gamma > 1,\ 1 > \eta > 0,\ \sigma_0 > 0,\ \epsilon_g > 0,\ \epsilon_E = \sqrt{L\epsilon_g}/3$ and $\epsilon_S = \epsilon_g/9$

for $k = 0, 1, \cdots,$ **do**

  evaluate $g_k = \nabla f(x_k),\ H_k = \nabla^2 f(x_k)$,

  compute an approximate eigenpair (α_k, v_k) such that $\alpha_k = v_k^\top H_k v_k \leqslant \lambda_{\min}(H_k) + \epsilon_E$

  if $\|g_k\| \leqslant \epsilon_g$ and $\alpha_k \geqslant -2\epsilon_E$ **then**

    return x_k

  end if

  if $\alpha_k \geqslant -\epsilon_E$ **then**

    solve the regularized subproblem approximately

$$s_k \approx \operatorname{argmin}_{s\in\mathbb{R}^n} \left\{ m_k^r(s) := g_k^\top s + \frac{1}{2} s^\top (H_k + 3\epsilon_E I)s + \frac{\sigma_k}{3}\|s\|^3 \right\}, \tag{9}$$

    $d_k = s_k$

  else

    solve the regularized subproblem approximately

$$s_k \approx \operatorname{argmin}_{s\in\mathbb{R}^n} \left\{ \tilde{m}_k^r(s) := g_k^\top s + \frac{1}{2} s^\top (H_k - \alpha_k I + 2\epsilon_E I)s + \tilde{J}_k(s) \right\}, \tag{10}$$

    where $\tilde{J}_k(s) = J_{\alpha_k,\sigma_k}(s)$

    if $\sigma_k\|s_k\| + \alpha_k \geqslant 0$ **then**

      $d_k = s_k$

    else

      $w_k = \beta v_k$ such that $\|w_k\| = |\alpha_k|$ and $w_k^\top g_k \leqslant 0$,

      $d_k = \frac{1}{2\sigma_k} w_k$

    end if

  end if

  $\rho_k = \frac{f(x_k) - f(x_k + d_k)}{-m_k(d_k)}$

  if $\rho_k \geqslant \eta$ or $\alpha_k < -\epsilon_E$ **then**

    $x_{k+1} = x_k + d_k,\ \sigma_{k+1} = \sigma_k/\gamma$ ▷ successful iteration

  else

    $x_{k+1} = x_k,\ \sigma_{k+1} = \gamma\sigma_k$ ▷ unsuccessful iteration

  end if

end for

where $\tilde{J}_k(s) = J_{\alpha_k,\sigma}(s) = \frac{\sigma}{3}\left[\max\left\{\|s\|, -\frac{\alpha_k}{\sigma}\right\}\right]^3 + \frac{\alpha}{2}\left[\max\left\{\|s\|, -\frac{\alpha_k}{\sigma}\right\}\right]^2$. Here we have $\sigma = \frac{L}{2}$ for Algorithm 1 and $\sigma = \sigma_k$ for Algorithm 2. Since our reformulation is designed for the case that the smallest eigenvalue of the Hessian is negative, we solve $m_k^r(s)$ when the approximate smallest eigenvalue is larger than or equal to criteria $-\epsilon_E$ and solve $\tilde{m}_k^r(s)$ otherwise.

To make algorithms more practical, we allow that the subproblems are approximately solved under certain criteria, i.e., the gradient norm of the model function is less than or equal to ϵ_S.

***Condition* 1** The subproblems (7) and (9) are approximately solved such that

$$\|\nabla m_k^r(s_k)\| \leqslant \epsilon_S. \tag{13}$$

The subproblems (8) and (10) are approximately solved such that

$$\|\nabla \tilde{m}_k^r(s_k)\| \leqslant \epsilon_S. \tag{14}$$

***Remark* 1** We may also replace Condition 1 by the following stopping criteria

$$\|\nabla m_k^r(s_k)\| \leqslant \max\left\{\zeta\|s_k\|^2, \epsilon_S\right\} \text{ and } \|\nabla \tilde{m}_k^r(s_k)\| \leqslant \max\left\{\zeta\|s_k\|^2, \epsilon_S\right\},$$

for some prescribed $\zeta \in (0, 1)$ where similar ideas are widely used in the literature [2, 4, 19]. Such stopping criteria have an advantage in practice if $\|s_k\|$ is large. By slightly modifying our proof, we still have an iteration complexity $\mathcal{O}(\epsilon_g^{-3/2})$ and an operation complexity $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$.

For simplicity of analysis, we consider the following condition for both Algorithms 1 and 2. We remark that the constants in the following condition may be changed slightly and we will still have the same order of complexity bounds.

***Condition* 2** Set $\epsilon_E = \frac{1}{3}\sqrt{L\epsilon_g}$ and $\epsilon_S = \frac{\epsilon_g}{9} = \frac{\epsilon_E^2}{L}$.

From now on, we suppose that Assumption 1 and Conditions 1 and 2 hold in the following of this paper. Our first main result is that both Algorithms 1 and 2 find an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point in at most $\mathcal{O}\left(\epsilon_g^{-3/2}\right)$ iterations (see Theorems 1 and 3). Then, we will show that under some mild assumptions (Assumptions 2 and 3), if the eigenvalue is approximated by the Lanczos procedure and the subproblem is approximately solved by NAG, then each iteration costs at most $\tilde{\mathcal{O}}(\epsilon_g^{-1/4})$ operations. Thus, the operation complexity of Algorithm 1 is $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ (see Theorem 2). Similar results also hold for the ARC and are omitted for simplicity.

***Remark* 2** Our goal is to present variants of the CR and ARC that are close to their practically efficient versions ([1, 2, 4]). Most of the existing works on the CR or ARC do not present an operation complexity $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ ([1, 2, 4, 19]), while other existing works in the framework of the CR or ARC that prove to admit an operation complexity $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ ([6, 11]) deviate more largely form the practically efficient versions than ours. The subproblem solver in [6] requires sophisticated parameter tuning and seems hard to implement in practice. The iteration number of each subproblem solver in [11] is set in advance, which may take additional cost in practice if the subproblem criteria are early met. Moreover, both works are restricted to the case of known gradient and/or Hessian Lipschitz constant, and they are restricted to the CR case. On the other hand, our methods are more close to the practically efficient CR and ARC algorithms in [1, 4]. We only add an additional regularizer $\frac{3}{2}\epsilon_E\|s\|^2$ or $\epsilon_E\|s\|^2$ to the original model function in the CR or ARC, use an approximate solution as the next step in most cases (in fact related to the easy case of the subproblem), and use a negative curvature direction in the other case (related to the hard case).

2.3 Progress in One Iteration of the Model Function

In this subsection, we give unified analysis for the descent progress in one iteration of the models for both the CR and ARC algorithms, which will be the heart of our convergence analysis of iteration complexity for the CR and ARC algorithms.

Proposition 2 *If Algorithm 1 terminates (at line 5) or Algorithm 2 terminates (at line 5), then the output x_k is an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point.*

Proof Note that in line 5 of either Algorithm 1 or Algorithm 2, we have $\|g_k\| \leqslant \epsilon_g$ and $\alpha_k \geqslant -2\epsilon_E$. Combining $\alpha_k \leqslant \lambda_{\min}(H_k) + \epsilon_E$ and $\alpha_k \geqslant -2\epsilon_E$, we obtain $\lambda_{\min}(H_k) \geqslant -3\epsilon_E = -\sqrt{L\epsilon_g}$ due to Condition 2. This, together with $\|g_k\| \leqslant \epsilon_g$, yields the desired result.

In the following two lemmas, we show sufficient decrease can be achieved in the case where either $\alpha_k \geqslant -\epsilon_E$ or $\alpha_k < -\epsilon_E$. The proofs for both lemmas defer to the appendix.

Lemma 1 *Suppose that x_k+d_k is not an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point. Suppose Assumption 1 and Conditions 1 and 2 hold and $\alpha_k \geqslant -\epsilon_E$ for some iteration k. Then, for Algorithm 1, we have*

$$-m(d_k) \geqslant \frac{\epsilon_E^3}{L^2}.$$

For Algorithm 2, we have

$$-m(d_k) \geqslant \min\left\{\frac{3}{\max\{\gamma, 2\sigma_0/L\}+1}, 1\right\} \cdot \frac{\epsilon_E^3}{L^2}.$$

Lemma 2 *Suppose that x_k+d_k is not an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point. Suppose Assumption 1 and Conditions 1 and 2 and in addition $1 < \gamma < 2$ for Algorithm 2, if $\alpha_k < -\epsilon_E$. Then, for Algorithm 1, we have*

$$-m(d_k) \geqslant \frac{\epsilon_E^3}{3L^2}.$$

For Algorithm 2, we have

$$-m(d_k) \geqslant \frac{\epsilon_E^3}{3(\max\{2\sigma_0, \gamma L\})^2}.$$

3 Convergence Analysis for the CR Algorithm

In this section, we first give iteration complexity analysis of the CR algorithm and then study its operation complexity in the case that the subproblem is solved by Nesterov's accelerated gradient method (NAG) and the approximate smallest eigenvalue of the Hessian is computed by the Lanczos procedure. The notation in this section follows that in Sect. 2.

We have the following theorem that gives a complexity bound that matches the best-known bounds in the literature [1, 4, 19].

Theorem 1 *Given Assumption 1 and Conditions 1 and 2, Algorithm 1 finds an* $(\epsilon_g, \sqrt{L\epsilon_g})$ *stationary point in at most* $\mathcal{O}\left(\epsilon_g^{-3/2}\right)$ *iterations.*

Proof First note that (12) implies $f(x_k + d_k) - f(x_k) \leqslant m(d_k)$. Combining Proposition 2, Lemmas 1 and 2 and Condition 2, and noting $\sigma = L/2$ for Algorithm 1, we have

$$f(x_k) - f(x_k + d_k) \geqslant \frac{1}{3L^2}\epsilon_E^3.$$

Adding the above inequalities from 0 to T, we have

$$f(x_0) - f(x_T) \geqslant \frac{T}{3L^2}\epsilon_E^3.$$

Noting that $f(x)$ is lower bounded from Assumption 1, we complete the proof.

Next, we give an estimation for the cost of each iteration and thus obtain the total operation complexity. Particularly, we invoke a backtracking line search version of NAG [18] (described in Algorithm 3) to approximately solve the subproblems (7) and (8) in Algorithm 1. Note that the objective functions m_k^r and $\tilde{m}_k^r$ in (7) and (8) are both ϵ_E-strongly convex. In Algorithm 3, h stands for either m_k^r or $\tilde{m}_k^r$.

Algorithm 3 NAG for minimizing m strongly convex smooth functions $h(z)$

Require $h, \nabla h, t_0 > 0, \theta_0 \in (0, 1], \beta \in (0, 1)$, initial point $z_0 \in \mathbb{R}^n$
for $l = 0, 1, \cdots$ **do**
  if $l \geqslant 1$ **then**
    $t_l = t_{l-1}$, ▷ initial step size for the lth iteration
    $\gamma_l = \frac{\theta_{l-1}^2}{t_{l-1}}$,
    $\frac{\theta_l^2}{t_l} = (1 - \theta_l)\gamma_l + m\theta_l$
  end if
  $y = z_l + \frac{\theta_l \gamma_l}{\gamma_l + m\theta_l}(v_l - z_l)$ ($y = z_0$ for $l = 0$),
  $z_{l+1} = y - t_l \nabla h(y)$
  while $h(y - t_l \nabla h(y)) > h(y) - \frac{t_l}{2}\|\nabla h(y)\|^2$ **do**
    $t_l = \beta t_l$,
    $z_{l+1} = y - t_l \nabla h(y)$
  end while
  $v_{l+1} = z_l + \frac{1}{\theta_l}(z_{l+1} - z_l)$
end for

Assumption 2 Suppose that for any extrapolated point y, and any $t \in (0, t_0]$, there exists an upper bound for $\|y - t\nabla h(y)\|$, i.e., there exists $M_n > 0$ such that $\|y - t\nabla h(y)\| \leqslant M_n$. Moreover, we assume

$$\|z_l\| \leqslant M_n, \ \forall l \geqslant 0 \text{ and } \|z^*\| \leqslant M_n,$$

where z_l is given in Algorithm 3 and z^* is the optimal solution of the subproblem (7) or (8).

The above assumption is easy to meet. Indeed, z_l is bounded because $h(z_l)$ is bounded from standard analysis for NAG (e.g., equation (15)), h is strongly convex and $\mathrm{dom}(h) = \mathbb{R}^n$, which is the case for m_k^r and $\tilde{m}_k^r$. Meanwhile, $y - t\nabla f(y)$ is bounded, if, noting that y is a linear combination of z_l and z_{l-1}, $\frac{\theta_l \gamma_l}{\gamma_l + m\theta_l}$ and $\frac{1}{\theta_l}$ are bounded constants, which is quite mild and holds in most practical cases.

We also make the following assumption that is widely used in the literature [2, 19].

Assumption 3 Suppose the Hessian H_k is bounded in each iteration of Algorithm 1, i.e., there exists some constant $M_H > 0$ such that

$$\|\nabla^2 f(x_k)\| \leqslant M_H.$$

The above two assumptions, together with Assumption 1, yield the following Lipschitz continuity result on the gradient $\nabla h(y)$.

Lemma 3 *Under Assumptions 1, 2 and 3, the gradients of m_k^r and $\tilde{m}_k^r$ are $L_S := 2M_H + 3\epsilon_E + LM_n$ Lipschitz continuous on the line path $[y, y - t_0\nabla h(y)]$ for any extrapolated point y in line 7 and the line path $[z_l, z^*]$ in Algorithm 3.*

Proof It suffices to show that for any p and q with $\|p\| \leqslant M_n$ and $\|q\| \leqslant M_n$, we have

$$\|\nabla h(p) - \nabla h(q)\| \leqslant L_S\|p - q\|,$$

where h stands for either m_k^r or $\tilde{m}_k^r$. From the definition of m_k^r, we have

$$\begin{aligned}
\|\nabla m_k^r(p) - \nabla m_k^r(q)\| &= \left\| H_k(p-q) + 3\epsilon_E(p-q) + \frac{L}{2}\|p\|p - \frac{L}{2}\|q\|q \right\| \\
&\leqslant \|H_k(p-q)\| + 3\epsilon_E\|p-q\| + \frac{L}{2}\|(\|p\|p - \|p\|q) + (\|p\|q - \|q\|q)\| \\
&\leqslant \left(\|H_k\| + 3\epsilon_E + \frac{L}{2}(\|p\| + \|q\|) \right) \|p-q\| \\
&\leqslant (M_H + 3\epsilon_E + LM_n)\|p-q\|,
\end{aligned}$$

where the last inequality follows from Assumptions 2 and 3.

To show the Lipschitz continuity of $\nabla \tilde{m}_k^r$, we need to consider three cases:

1. Both $\|p\| + \frac{2\alpha_k}{L} \geqslant 0$ and $\|q\| + \frac{2\alpha_k}{L} \geqslant 0$. In this case, both $\tilde{\nabla} m_k^r(p) = \nabla m_k^r(p) - \epsilon_E p$ and $\tilde{\nabla} m_k^r(q) = \nabla m_k^r(q) - \epsilon_E q$. With a similar analysis to the previous proof, it is easy to show $\tilde{\nabla} m_k^r(p)$ is $(M_H + 2\epsilon_E + LM_n)$ Lipschitz continuous.
2. Both $\|p\| + \frac{2\alpha_k}{L} \leqslant 0$ and $\|q\| + \frac{2\alpha_k}{L} \leqslant 0$. It is trivial to see $\tilde{\nabla} m_k^r(p)$ is $(M_H + 2\epsilon_E)$ Lipschitz continuous as $\nabla \tilde{J}_k(p) = \nabla \tilde{J}_k(q) = 0$.
3. Either (i) $\|p\| + \frac{2\alpha_k}{L} > 0$, $\|q\| + \frac{2\alpha_k}{L} \leqslant 0$ or (ii) $\|p\| + \frac{2\alpha_k}{L} \leqslant 0$, $\|q\| + \frac{2\alpha_k}{L} > 0$. Due to symmetry, we only prove the first case. From Proposition 1, we have

$$\begin{aligned}
\|\nabla \tilde{m}_k^r(p) - \nabla \tilde{m}_k^r(q)\| &= \left\| (H_k - \alpha_k I)(p-q) + 2\epsilon_E(p-q) + \left(\frac{L}{2}\|p\| + \alpha_k\right) p - 0 \right\| \\
&\leqslant \|(H_k - \alpha_k I)(p-q)\| + 2\epsilon_E\|p-q\| + \left\| (\frac{L}{2}\|p\| + \alpha_k) p \right\| \\
&\leqslant (2\|H_k\| + 3\epsilon_E)\,\|p-q\| + \left\| \left(\frac{L}{2}\|p\| + \alpha_k\right) p - \left(\frac{L}{2}\|q\| + \alpha_k\right) p \right\| \\
&\leqslant \left(2M_H + 3\epsilon_E + \frac{L}{2} M_n\right) \|p-q\|,
\end{aligned}$$

where in the second inequality we use $\|H_k - \alpha_k I\| \leqslant 2\|H_k\| + \epsilon_E$ as $\lambda_{\min}(H_k) + \epsilon_E \geqslant \alpha_k \geqslant \lambda_{\min}(H_k)$ and $2L\|q\| + \alpha_k \leqslant 0$, and the last inequality follows from Assumptions 2 and 3.

Now let us give an estimation for the iteration complexity of Algorithm 3 to achieve a point such that $\|\nabla h(z_l)\| \leqslant \epsilon_S$.

Lemma 4 *Suppose Algorithm 3 is used as subproblem solvers for (7) and (8). Given Conditions 1 and 2 and Assumptions 1, 2 and 3, Algorithm 3 takes at most* $\tilde{\mathcal{O}}\left(\epsilon_E^{-1/2}\right)$ *iterations to achieve a point such that* $\|\nabla h(z_l)\| \leqslant \epsilon_S = \frac{\epsilon_E^2}{L}$. *Moreover, the cost in each iteration is dominated by two matrix vector products.*

Proof Note that either m_k^r or $\tilde{m}_k^r$ is ϵ_E-strongly convex due to the definitions, and L_S-smooth due to Lemma 3. From complexity results of NAG in [18, 20], we obtain that

$$h(z_l) - h^* \leqslant \Pi_{i=1}^{l-1}(1 - \sqrt{\epsilon_E t_i})C, \tag{15}$$

where $C = \left((1-\theta_0)(h(x_0) - h^*) + \frac{\theta_0^2}{2t_0}\|x_0 - x^*\|^2\right)$ and $t_i \geqslant \min\{t_0, \beta/L_S\}$. Thus, (15) further yields

$$h(z_l) - h^* \leqslant \left(1 - \sqrt{\epsilon_E \min\{t_0, \beta/L_S\}}\right)^{k-1} C.$$

Therefore, it takes at most $T = \mathcal{O}\left(\sqrt{\frac{1}{\epsilon_E}} \log \frac{1}{\epsilon_h}\right)$ to achieve a solution such that $h(z_T) - h^* \leqslant \epsilon_h$.

From the L_S smoothness of m_k^r and $\tilde{m}_k^r$ along the line $[z_l, z^*]$ (due to Lemma 3), we further have

$$\frac{1}{2L_S}\|\nabla h(z_l)\|^2 \leqslant h(z_l) - h^*, \quad \forall k \geqslant 0.$$

Thus, by letting $\epsilon_h = \epsilon_S^2/2L_S$, we have

$$\|\nabla h(z_l)\| \leqslant \sqrt{2L_S\epsilon_h} = \epsilon_S = \frac{1}{9}\epsilon_g = \frac{\epsilon_E^2}{L}.$$

Hence, the iteration complexity for $\|\nabla h(z_T)\| \leqslant \epsilon_S$ is $\mathcal{O}\left(\sqrt{\frac{1}{\epsilon_E}}\log\frac{1}{\epsilon_E}\right) = \tilde{\mathcal{O}}\left(\epsilon_E^{-1/2}\right)$.

Note that each iteration of Algorithm 3 requires one gradient evaluation of $\nabla h(y)$ according to the expression of m_k^r and $\tilde{m}_k^r$, where the most expensive operator is the Hessian-vector product $H_k y$. Then, the function evaluation of $h(y)$ is cheap if we store $H_k y$. Meanwhile, to compute $m_k(y - t_l\nabla h(y))$ for different t_l, we have

$$\begin{aligned} m_k(y - t_l\nabla h(y)) &= g_k^\top y - t_l g_k^\top \nabla h(y) + \frac{1}{2}y^\top H_k y - t_l y^\top H_k \nabla h(y) \\ &\quad + \frac{t_l^2}{2}\nabla h(y)^\top H_k \nabla h(y) + \frac{L}{6}\|y - t_l\nabla h(y)\|^3, \end{aligned}$$

which costs $\mathcal{O}(1)$ if $H_k\nabla h(y),\ g_k^\top y,\ g_k^\top \nabla h(y),\ y^\top H_k y,\ y^\top H_k \nabla h(y),\ \|y\|,\ y^\top \nabla h(y)$ and $\|\nabla h(y)\|$ are provided (using $\|y - t_l\nabla h(y)\|^2 = \|y\|^2 - 2t_l y^\top \nabla h(y) + \|t_l\nabla h(y)\|^2$). Note that in the lth iteration, we have $t_0 \geqslant t_l \geqslant \min\{\beta/L_S, t_0\}$. We thus at most do $\mathcal{O}(1)$ searches for β. So in one iteration, the total cost is two matrix vectors products and $\mathcal{O}(n)$ other operations. With a similar analysis, the same complexity result holds for $\tilde{m}_k^r$.

The following lemma shows a well-known result that the smallest eigenvalue of a given matrix can be computed efficiently with high probability.

Lemma 5 ([21] *and Lemma 9 in* [8]) *Let H be a symmetric matrix satisfying $\|H\| \leqslant U_H$ for some $U_H > 0$, and $\lambda_{\min}(H)$ its minimum eigenvalue. Suppose that the Lanczos procedure is applied to find the largest eigenvalue of $U_H I - H$ starting at a random vector distributed uniformly over the unit sphere. Then, for any $\epsilon > 0$ and $\delta \in (0, 1)$, there is a probability at least $1 - \delta$ that the procedure outputs a unit vector v such that $v^\top H v \leqslant \lambda_{\min}(H) + \epsilon$ in at most $\min\left\{n, \frac{\log(n/\delta^2)}{2\sqrt{2}}\sqrt{\frac{U_H}{\epsilon}}\right\}$ iterations.*

Now we are ready to present the main result in this section that Algorithm 1 has an operation complexity $\tilde{\mathcal{O}}\,(\epsilon_g^{-7/4})$.

Theorem 2 *Suppose the approximate eigenpair in line 3 of Algorithm 1 is computed by the Lanczos Procedure, and subproblems (7) and (8) are approximately solved by Algorithm 3. Under Conditions 1 and 2 and Assumptions 1, 2 and 3, the algorithm finds an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point with high probability, and in this case the operation complexity of Algorithm 1 is $\tilde{\mathcal{O}}\,(\epsilon_g^{-7/4})$.*

Proof First, note that the iteration complexity is $\mathcal{O}(\epsilon_g^{-3/2})$, due to Theorem 1.

At each iteration, if the subproblems are approximately solved in line 8 or 11 in Algorithm 1, the subproblem iteration complexity is $\tilde{\mathcal{O}}(\epsilon_E^{-1/2}) = \tilde{\mathcal{O}}(\epsilon_g^{-1/4})$ because that $\epsilon_E = \sqrt{L\epsilon_g}/3$, and that the dominated cost is $\tilde{\mathcal{O}}(\epsilon_g^{-1/4})$ matrix vector products, thanks to Lemma 4.

Another cost at each iteration is inexactly computing the smallest eigenvalue. Note that the failure probability of the Lanczos procedure is only in the "log factor" in the complexity bound. Hence, for any given $\delta' \in (0, 1)$, in the Lanczos procedure we can use a very small δ like $\delta = \delta'/T$, where T is the total iteration number bounded by $\mathcal{O}(\epsilon_g^{-3/2})$. Then, from the union bound, the full Algorithm 1 finds an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point with probability $1-\delta'$. From Lemma 5, it takes $\tilde{\mathcal{O}}(\epsilon_E^{-1/2}) = \tilde{\mathcal{O}}(\epsilon_g^{-1/4})$ matrix vector products to achieve an ϵ_E approximate eigenpair, with probability at least $1 - \delta'$.

As the iteration complexity of Algorithm 1 is $\mathcal{O}(\epsilon_g^{-3/2})$ and each iteration takes $\tilde{\mathcal{O}}(\epsilon_g^{-1/4})$ unit operations, we conclude that the operation complexity is $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$.

4 Convergence Analysis for the ARC Algorithm

In this section, we first show that the ARC algorithm also has an iteration complexity $\mathcal{O}\left(\epsilon_g{}^{-3/2}\right)$ for finding an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point. Then, we will briefly analyze its operation complexity in the case that the subproblem is solved by NAG and the approximate smallest eigenvalue of the Hessian is computed by the Lanczos procedure. The notation in this section follows that in Sect. 2.

To show the iteration complexity of the ARC algorithm is still $\mathcal{O}(\epsilon_g^{-3/2})$, the key proof here is that we need to counter the iteration number for successful steps. Specifically, we need the following lemma that shows when σ_k is large enough, the iteration must be successful.

Lemma 6 *Suppose Assumption 1 holds. If $\sigma_k \geqslant L/2$ and $m_k(d_k) < 0$, then the kth iteration is successful.*

Proof By (12) and $\sigma_k \geqslant L/2$, we have

$$\begin{aligned} f(x_k + d_k) - f(x_k) &\leqslant g_k^\top d_k + \frac{1}{2} d_k^\top H_k d_k + \frac{L}{6}\|d_k\|^3 \\ &\leqslant g_k^\top d_k + \frac{1}{2} d_k^\top H_k d_k + \frac{\sigma_k}{3}\|d_k\|^3 \\ &= m_k(d_k) < 0. \end{aligned}$$

This yields $\rho_k = \frac{f(x_k) - f(x_k + d_k)}{-m_k(d_k)} \geqslant 1 > \eta$. Thus, the kth iteration is successful.

The following lemma shows that the adaptive regularizer is bounded above.

Lemma 7 *Suppose Assumption 1 holds. Then, $\sigma_k \leqslant \max\{\sigma_0, \gamma L/2\}$, $\forall k \geqslant 0$.*

Proof Suppose the kth iteration is the first unsuccessful iteration such that $\sigma_{k+1} = \gamma\sigma_k \geqslant \gamma L/2$, which implies $\sigma_k \geqslant L/2$. However, from Lemma 6, we know that the kth iteration must be successful and thus $\sigma_{k+1} = \sigma_k/\gamma < \sigma_k$, which is a contradiction.

Now we are ready to present our main convergence result of Algorithm 2, which is of the same order with the best-known iteration bound ([4, 19]).

Theorem 3 *Suppose that Assumption 1 and Conditions 1 and 2 hold, and* $\max\{\frac{\sigma_0}{L}, \frac{\gamma}{2}\} \leqslant 1$. *Then, Algorithm 2 takes* $T \leqslant \mathcal{O}(\epsilon_g^{-3/2})$ *iterations to find an* $(\epsilon_g, \sqrt{L\epsilon_g})$ *stationary point.*

Proof Note that

$$T = |\mathcal{S}| + |\mathcal{U}|, \tag{16}$$

where $\mathcal{S}$ is the index set of successful iterations and $\mathcal{U}$ is the index set of unsuccessful iterations. Here, $|\mathcal{A}|$ denotes the cardinality of a set $\mathcal{A}$. Since $\sigma_T = \sigma_0\gamma^{|\mathcal{U}|-|\mathcal{S}|}$ and $\sigma_T \leqslant \max\{\sigma_0, \gamma L/2\}$ due to Lemma 7, we have

$$|\mathcal{U}| \leqslant \max\left\{0, \log_\gamma\left(\frac{\gamma L}{2\sigma_0}\right)\right\} + |\mathcal{S}|. \tag{17}$$

Note also that $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$, where

$$\begin{aligned}
\mathcal{S}_1 &:= \{k \in \mathcal{S} : \|\nabla f(x_k + d_k)\| \leqslant \epsilon_g \text{ and } \lambda_{\min}(H_{k+1}) \geqslant -\sqrt{L\epsilon_g}\},\\
\mathcal{S}_2 &:= \mathcal{S} \setminus \mathcal{S}_1.
\end{aligned}$$

Now we have

$$\begin{aligned}
f(x_0) - f^* &\geqslant \sum_{k=0}^{\infty} f(x_k) - f(x_{k+1}) = \sum_{k\in\mathcal{S}} f(x_k) - f(x_{k+1})\\
&\geqslant \sum_{k\in\mathcal{S}_2} f(x_k) - f(x_{k+1})\\
&\geqslant \sum_{k\in\mathcal{S}_2} -\eta m_k(d_k)\\
&\geqslant \sum_{k\in\mathcal{S}_2} \eta \min\left\{\frac{3}{\max\{\gamma, 2\sigma_0/L\}+1}, 1, \frac{1}{3(\max\{2\sigma_0/L, \gamma\})^2}\right\} \cdot \frac{\epsilon_E^3}{L^2}
\end{aligned}$$

where the fifth inequality follows from Lemmas 1 and 2. This, together with $\epsilon_E = \sqrt{L\epsilon_g}/3$, gives

$$|\mathcal{S}_2| \leqslant \mathcal{O}(\epsilon_g^{-3/2}).$$

It is obvious that $|\mathcal{S}_1| = 1$ as the algorithm terminates in one iteration. Then, we have

$$|\mathcal{S}| = |\mathcal{S}_1| + |\mathcal{S}_2| \leqslant \mathcal{O}\left(\epsilon_g^{-3/2}\right).$$

This, together with (16) and (17), gives $T \leqslant \mathcal{O}(\epsilon_g^{-3/2})$.

In fact, with a similar analysis to Sect. 3, we can show that the operation complexity for Algorithm 2 is still $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ to find an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point under mild conditions with high probability, if NAG and the Lanczos procedure are used in each iteration. This is because the matrix vector product number in each iteration of Algorithm 2 is still $\tilde{\mathcal{O}}(\epsilon_g^{-1/4})$. Two key observations for proving the $\tilde{\mathcal{O}}(\epsilon_g^{-1/4})$ bound of NAG are that σ_k is upper bounded by constants as shown in Theorem 3, and that the subproblems are still ϵ_E-strongly convex and Lipschitz smooth. The Lipschitz smoothness follows from a similar technique with Lemma 3 under Assumptions 2 and 3.

5 Numerical Experiments

This section mainly shows the effects of our new subproblem reformulation without the additional regularizer $\epsilon_E\|s\|^2$ for the ARC algorithm. We did numerical experiments among ARC algorithms ([2]) with different subproblem solvers and compared their performance. We point out that we do not directly implement Algorithm 2 since it is practically inefficient if we compute the minimum eigenvalue of the Hessian at every iteration. Particularly, in Algorithm 4, we only call a subproblem solver based on reformulation (CRS_u) if a prescribed condition is met.

Let f denote the objective function, g_k denote the gradient $\nabla f(x_k)$ and H_k denote the Hessian $\nabla^2 f(x_k)$. In Algorithm 4, we use the Cauchy point s_k^C (as in [2]) as the initial point of the subproblem solver in each iteration:

$$s_k^C = -\alpha_k^C g_k \text{ and } \alpha_k^C = \operatorname*{argmin}_{\alpha\in\mathbb{R}_+} m_k(-\alpha g_k),$$

which is obtained by globally minimizing $m_k(s) = g_k^s + s^\top H_k s + \frac{\sigma_k}{3}\|s\|^3$ along the current negative gradient direction. Let $\mathcal{A}$ denote an arbitrary solver for (CRS), $\mathcal{A}_r$ denote an arbitrary solver for the constrained reformulation (CRS_r) and $\mathcal{A}_u$ denote an arbitrary solver for the unconstrained reformulation (CRS_u). Because the subproblem solver $\mathcal{A}_u$ (or $\mathcal{A}_r$) are designed for cases where H_k is not positive semidefinite, and the Cauchy point is a good initial point when the norm of the gradient is large, we call the solver $\mathcal{A}_u$ (or $\mathcal{A}_r$) if the following condition is met:

$$\|g_k\| \leqslant \max\left(f(x_k), 1\right)\cdot\epsilon_1 \quad \text{and} \quad \lambda_{\min}(H_k) < -\epsilon_2, \tag{18}$$

where ϵ_1 and ϵ_2 are some small positive real numbers and $\lambda_{\min}(H_k)$ is the minimum eigenvalue of H_k. If condition (18) is not met, we call $\mathcal{A}$ to solve the model function directly. We only accept the (approximate) solution s_k if $m_k(s_k)$ is smaller than that $m_k(s_k^C)$; otherwise the Cauchy point s_k^C is used. This guarantees that Algorithm 4 converges to a first-order stationary point under mild conditions ([2,Lemma 2.1]).

Algorithm 4 ARC using convex reformulation

Require $x_0,\ \gamma_2 \geqslant \gamma_1 > 1,\ 1 > \eta_2 \geqslant \eta_1 > 0$, and $\sigma_0 > 0$ for $k = 0, 1, \cdots$ until convergence

compute the Cauchy point s_k^C
if condition (18) is satisfied **then**
    compute a trial step $\bar{s}_k$ using $\mathcal{A}_u$ (or $\mathcal{A}_r$) with an initial point s_k^C
else
    compute a trial step $\bar{s}_k$ using $\mathcal{A}$ with an initial point s_k^C
end if
set

$$s_k = \begin{cases} \bar{s}_k, & \text{if } m_k(\bar{s}_k) \leqslant m_k(s_k^C); \\ s_k^C, & \text{otherwise.} \end{cases}$$

compute $f(x_k + s_k)$ and

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{-m_k(s_k)},$$

set

$$x_{k+1} = \begin{cases} x_k + s_k, & \text{if } \rho_k \geqslant \eta_1; \\ x_k, & \text{otherwise.} \end{cases}$$

set

$$\sigma_{k+1} \in \begin{cases} (0, \sigma_k], & \text{if } \rho_k > \eta_2; & \text{(very successful iteration)} \\ [\sigma_k, \gamma_1 \sigma_k], & \text{if } \eta_1 \leqslant \rho_k \leqslant \eta_2; & \text{(successful iteration)} \\ [\gamma_1 \sigma_k, \gamma_2 \sigma_k], & \text{otherwise.} & \text{(unsuccessful iteration)} \end{cases}$$

We experimented with two subproblem solvers $\mathcal{A}_u$ for Algorithm 4. The first one is the gradient method with Barzilai–Borwein step size ([22]) and the second one is NAG (here we denote it by APG to keep consistent with [13]). More specifically, in our implementation, if condition (18) is not satisfied, we still solve (CRS) by BBM; otherwise we implement BBM or APG to solve the unconstrained problem (CRS_u). The former is termed `ARC-URBB`, while the latter is termed `ARC-URAPG`. We compare our algorithms to the ARC algorithm in [2], denoted by `ARC-GLRT`, in which the subproblems are solved by the generalized Lanczos method. Besides, we also implement Algorithm 4 with two different subproblem solvers $\mathcal{A}_r$ in [13], denoted by `ARC-RBB` and `ARC-RAPG`, in which the subproblems are reformulated as (CRS_r) and solved by BBM and APG, respectively.

We implemented all the ARC algorithms in MATLAB R2017a on a Macbook Pro laptop with 4 Intel i5 cores (1.4GHz) and 8GB of RAM. The implementations are based on 20 medium-size ($n \in [500, 1500]$) problems from the CUTEst collections ([23]) as in [13], where condition (18) is satisfied in at least one iteration in our new algorithm. For condition (18), we set $\epsilon_1 = 10^{-2}$ and $\epsilon_2 = 10^{-4}$. Other parameters in ARC are chosen as described in [2]. All the subproblem solvers use the same eigenvalue tolerance, stopping criteria, and initialization as in [13]. For BBMs, a simple line search rule is used to guarantee the decrease in the objective function values. For APGs, a well-known restarting strategy ([24, 25]) is used to speed up the algorithm.

The numerical results are reported in Table 1. The first column indicates the name of the problem instance with its dimension. The column f^*, n_i, n_{prod} , n_f , n_g and n_{eig} show the final objective value, the iteration number, number of Hessian-vector products, number of function evaluations, number of gradient evaluations and the number of eigenvalue computations. The columns time, time$_{\text{eig}}$ and time$_{\text{loop}}$, show in seconds the overall CPU time, eigenvalue computation time and difference between the last two, respectively. Each value is an average of 10 realizations with different initial points. Table 1 shows that with the same stopping criteria, all algorithms return the same objective function value on 18 of the problems, except ARC-RAPG, ARC-URBB and ARC-URAPG on the problem BROYDN7D with a lower final objective function value, and ARC-GLRT on the problem CHAINWOO with a lower final objective function value. Table 1 also shows the quantities n_i, n_{prod}, n_f and n_g of the five algorithms are similar. For several problems, ARC-URBB and ARC-URAPG based on our new reformulation have some advantages on n_{prod}. Due to the eigenvalue calculation, four algorithms based on the convex reformulation require additional manipulation, resulting in a larger total CPU time, evidenced by the column time, which was also observed in [13]. The column time$_{\text{loop}}$ shows that all the algorithms have a similar CPU time if we exclude the time for computing the eigenvalues.

To investigate the numerical results more clearly, we illustrate the experiments by performance profiles (Figs. 1, 2, 3) ([26]). According to the performs profiles, although ARC-GLRT has the best performance, the iteration numbers and the gradient evaluation numbers of ARC-URBB and ARC-URAPG are less than 2 times of those by ARC-GLRT on over 95% of the tests, and Hessian-vector product number of ARC-URBB is less than 2 times of those by ARC-GLRT on about 85% of the tests. Noting that ARC-URBB, ARC-URAPG, ARC-RBB and ARC-RAPG have the similar performance, we thus plot the performance profiles on test problems for these 4 algorithms in Figs. 4, 5 and 6. We find ARC-URAPG has the best iteration number and gradient evaluation number, and both ARC-URBB and ARC-URAPG have better Hessian-vector product number.

We also investigate the numerical results for all 10 implementations with different initializations, in order to show the advantages of the new algorithms more comprehensively. Table 2 reports the number that ARC-URBB or ARC-URAPG outperforms ARC-GLRT, ARC-RBB and ARC-RAPG out of the 10 realizations for each problem. It shows our algorithms frequently outperform ARC-GLRT, ARC-RBB and ARC-RAPG in iteration number, number of Hessian-vector products and gradient evaluations.

Table 1 Results on the CUTEst problems

Problem	Method	n_i	n_{prod}	n_f	n_g	n_{eig}	f^*	Time	Time_{eig}	$\text{Time}_{\text{loop}}$
BROYDN7D	ARC-GLRT	42.7	828.9	43.7	35.6	–	2.42e+02	0.340	–	0.340
(1000)	ARC-RBB	43.6	946.9	44.6	36.3	17.4	2.40e+02	1.071	0.681	0.390
	ARC-RAPG	43.7	933.5	44.7	36.2	17.9	2.39e+02	1.100	0.701	0.399
	ARC-URBB	43.8	933.2	44.8	36.4	17.9	2.39e+02	1.059	0.688	0.371
	ARC-URAPG	43.4	845.8	44.4	35.9	17.2	2.39e+02	1.058	0.664	0.394
BRYBND	ARC-GLRT	34.3	1 575.5	35.3	29.3	–	2.73e+01	0.487	–	0.487
(1000)	ARC-RBB	29.8	1 314.3	30.8	25.7	6.5	2.73e+01	1.352	0.946	0.406
	ARC-RAPG	29.8	1 288.9	30.8	25.7	6.5	2.73e+01	1.426	1.005	0.421
	ARC-URBB	29.8	1 278.7	30.8	25.7	6.5	2.73e+01	1.349	0.956	0.394
	ARC-URAPG	29.8	1 262.1	30.8	25.7	6.5	2.73e+01	1.376	0.975	0.402
CHAINWOO	ARC-GLRT	203.5	5 462.5	204.5	152.7	–	1.07e+03	1.576	–	1.576
(1000)	ARC-RBB	293.1	10 705.2	294.1	218.7	172.5	1.17e+03	26.495	23.215	3.280
	ARC-RAPG	299.5	10 625.8	300.5	225.4	178.4	1.17e+03	27.363	23.841	3.522
	ARC-URBB	291.4	8 363.3	292.4	219.4	168.0	1.17e+03	26.276	23.553	2.723
	ARC-URAPG	303.4	8 683.5	304.4	227.9	178.9	1.16e+03	26.534	23.434	3.099
DIXMAANF	ARC-GLRT	23.8	599.6	24.8	22.6	–	1.00e+00	0.421	–	0.421
(1500)	ARC-RBB	22.1	572.6	23.1	21.2	10.1	1.00e+00	1.391	0.964	0.427
	ARC-RAPG	22.2	543.8	23.2	21.1	10.2	1.00e+00	1.391	0.969	0.423
	ARC-URBB	22.6	535.3	23.6	21.5	10.6	1.00e+00	1.415	1.009	0.406
	ARC-URAPG	22.3	477.2	23.3	21.2	10.3	1.00e+00	1.385	0.974	0.412
DIXMAANG	ARC-GLRT	24.9	606.7	25.9	23.0	–	1.00e+00	0.413	–	0.413
(1500)	ARC-RBB	24.6	652.8	25.6	22.6	11.0	1.00e+00	1.446	0.982	0.464
	ARC-RAPG	23.7	597.5	24.7	22.2	10.1	1.00e+00	1.378	0.927	0.451
	ARC-URBB	23.0	418.1	24.0	21.9	9.8	1.00e+00	1.270	0.912	0.358
	ARC-URAPG	23.3	441.9	24.3	22.0	10.0	1.00e+00	1.274	0.902	0.372
DIXMAANH	ARC-GLRT	29.6	680.8	30.6	25.9	–	1.00e+00	0.461	–	0.461
(1500)	ARC-RBB	30.7	696.6	31.7	26.2	13.3	1.00e+00	1.705	1.186	0.519
	ARC-RAPG	30.5	664.4	31.5	26.1	13.1	1.00e+00	1.696	1.189	0.507
	ARC-URBB	30.5	625.3	31.5	26.0	13.1	1.00e+00	1.659	1.178	0.480
	ARC-URAPG	30.5	619.2	31.5	26.0	13.4	1.00e+00	1.681	1.198	0.483
DIXMAANJ	ARC-GLRT	43.7	4 519.5	44.7	37.6	–	1.00e+00	2.311	–	2.311
(1500)	ARC-RBB	48.7	2 952.9	49.7	42.4	30.3	1.00e+00	33.409	31.727	1.682
	ARC-RAPG	51.1	3 324.3	52.1	43.5	33.1	1.00e+00	37.646	35.873	1.774
	ARC-URBB	50.1	2 937.5	51.1	43.5	32.4	1.00e+00	35.312	33.738	1.574
	ARC-URAPG	49.2	2 743.4	50.2	42.7	31.3	1.00e+00	33.913	32.392	1.521
DIXMAANK	ARC-GLRT	51.1	4 883.9	52.1	43.2	–	1.00e+00	2.458	–	2.458
(1500)	ARC-RBB	63.1	4 382.3	64.1	53.1	42.9	1.00e+00	40.483	38.208	2.275
	ARC-RAPG	63.9	4 453.5	64.9	53.3	43.8	1.00e+00	41.436	39.114	2.322
	ARC-URBB	60.7	3 523.5	61.7	51.5	41.1	1.00e+00	39.471	37.603	1.868
	ARC-URAPG	62.4	3 962.9	63.4	52.5	42.4	1.00e+00	40.038	37.941	2.097

Table 1 continued

Problem	Method	n_i	n_{prod}	n_f	n_g	n_{eig}	f^*	Time	Time_{eig}	$\text{Time}_{\text{loop}}$
DIXMAANL	ARC-GLRT	57.7	4 569.5	58.7	47.6	–	1.00e+00	2.334	–	2.334
(1500)	ARC-RBB	65.2	4 126.9	66.2	55.0	40.9	1.00e+00	40.609	38.454	2.155
	ARC-RAPG	66.0	4 103.8	67.0	55.1	42.1	1.00e+00	40.438	38.255	2.183
	ARC-URBB	61.3	3 398.6	62.3	52.2	37.3	1.00e+00	37.372	35.571	1.801
	ARC-URAPG	65.4	3 721.6	66.4	54.6	41.3	1.00e+00	39.836	37.842	1.995
EXTROSNB	ARC-GLRT	1 824.2	54 022.6	1 825.2	1 274.9	–	1.47e-08	16.641	–	16.641
(1000)	ARC-RBB	1 344.0	192 873.0	1 345.0	1 094.9	1 236.4	2.99e-06	72.713	12.047	60.666
	ARC-RAPG	1 341.0	192 160.7	1 342.0	1 097.3	1 234.1	2.99e-06	71.442	11.863	59.579
	ARC-URBB	1 383.9	198 543.3	1 384.9	1 129.2	1 276.2	2.99e-06	73.391	12.177	61.215
	ARC-URAPG	1 397.9	200 543.8	1 398.9	1 121.8	1 291.3	2.98e-06	73.764	12.293	61.471
FLETCHCR	ARC-GLRT	1 969.9	42 563.3	1 970.9	1 327.2	–	1.20e+00	12.710	–	12.710
(1000)	ARC-RBB	1 982.0	53 970.1	1 983.0	1 357.0	774.8	1.20e+00	38.237	13.914	24.324
	ARC-RAPG	1 984.0	53 642.5	1 985.0	1 368.1	787.8	1.20e+00	38.062	13.562	24.500
	ARC-URBB	1 980.8	52 054.4	1 981.8	1 365.5	782.0	1.20e+00	38.458	14.402	24.056
	ARC-URAPG	1 976.8	51 214.6	1 977.8	1 361.9	771.3	1.20e+00	38.390	14.234	24.156
FREUROTH	ARC-GLRT	36.7	366.1	37.7	30.3	–	1.17e+05	0.302	–	0.302
(1000)	ARC-RBB	33.8	1 122.4	34.8	30.2	21.2	1.17e+05	0.697	0.205	0.492
	ARC-RAPG	36.0	1 371.1	37.0	31.6	23.5	1.17e+05	0.787	0.222	0.565
	ARC-URBB	36.5	1 400.4	37.5	30.2	24.0	1.17e+05	0.885	0.247	0.639
	ARC-URAPG	34.8	1 207.6	35.8	30.1	22.0	1.17e+05	0.810	0.219	0.591
GENHUMPS	ARC-GLRT	1 702.9	50 838.9	1 703.9	1 039.5	–	8.73e-13	15.912	–	15.912
(1000)	ARC-RBB	1 525.5	41 837.4	1 526.5	922.5	9.3	7.06e-12	20.876	0.206	20.670
	ARC-RAPG	1525.4	41 841.4	1 526.4	922.4	9.3	8.90e-12	19.249	0.218	19.030
	ARC-URBB	1 525.5	41 729.6	1 526.5	922.6	9.3	8.34e-12	22.510	0.199	22.311
	ARC-URAPG	1 525.4	41 762.7	1 526.4	922.5	9.3	1.44e-11	21.270	0.200	21.071
GENROSE	ARC-GLRT	1 058.6	20 703.8	1 059.6	711.7	–	1.00e+00	2.818	–	2.818
(500)	ARC-RBB	1 079.7	28 236.6	1 080.7	736.9	166.5	1.00e+00	4.092	0.944	3.149
	ARC-RAPG	1 151.5	29 887.7	1 152.5	780.7	191.6	1.00e+00	3.594	0.863	2.732
	ARC-URBB	1 081.1	28 124.1	1 082.1	737.5	164.3	1.00e+00	3.848	0.890	2.958
	ARC-URAPG	1 083.4	27 979.1	1 084.4	737.1	169.7	1.00e+00	4.268	1.005	3.263
NONCVXU2	ARC-GLRT	65.5	8 065.7	66.5	61.5	–	2.32e+03	2.083	–	2.083
(1000)	ARC-RBB	127.5	7 660.5	128.5	122.1	124.5	2.32e+03	78.082	75.564	2.518
	ARC-RAPG	122.4	7 637.8	123.4	118.9	119.6	2.32e+03	77.664	75.163	2.501
	ARC-URBB	123.4	7 845.9	124.4	119.6	120.6	2.32e+03	78.858	76.286	2.571
	ARC-URAPG	113.8	7 211.2	114.8	109.9	111.0	2.32e+03	71.320	69.111	2.209
NONCVXUN	ARC-GLRT	300.9	224 970.5	301.9	294.5	–	2.32e+03	49.239	–	49.239
(1000)	ARC-RBB	2 025.5	283 403.7	2 026.5	2 018.8	2 021.6	2.32e+03	1 414.201	1 346.432	67.769
	ARC-RAPG	2 116.6	295 837.9	2 117.6	2 109.8	2 112.9	2.32e+03	1 479.271	1 407.690	71.581
	ARC-URBB	2 483.0	350 646.4	2 484.0	2 477.1	2 479.3	2.32e+03	1 734.238	1 650.711	83.527
	ARC-URAPG	2 105.2	294 676.3	2 106.2	2 098.3	2 101.3	2.32e+03	1 466.452	1 393.276	73.176

Table 1 continued

Problem	Method	n_i	n_{prod}	n_f	n_g	n_{eig}	f^*	Time	Time$_{\text{eig}}$	Time$_{\text{loop}}$
OSCIPATH	ARC-GLRT	39.3	6 079.9	40.3	31.3	–	3.12e-01	0.516	–	0.516
(500)	ARC-RBB	56.4	5 658.1	57.4	49.5	27.8	3.12e-01	5.502	5.204	0.298
	ARC-RAPG	57.0	5 747.5	58.0	49.6	28.1	3.12e-01	5.702	5.348	0.354
	ARC-URBB	58.7	6 007.2	59.7	52.0	30.5	3.12e-01	6.163	5.845	0.318
	ARC-URAPG	57.3	5 799.2	58.3	50.8	28.7	3.12e-01	5.866	5.532	0.334
TOINTGSS	ARC-GLRT	19.2	118.6	20.2	14.1	–	1.00e+01	0.119	–	0.119
(1000)	ARC-RBB	15.4	368.2	16.4	12.2	10.3	1.00e+01	0.269	0.086	0.183
	ARC-RAPG	15.9	494.7	16.9	12.4	10.9	1.00e+01	0.296	0.081	0.215
	ARC-URBB	15.0	322.9	16.0	11.8	10.0	1.00e+01	0.233	0.076	0.156
	ARC-URAPG	15.6	372.6	16.6	12.5	10.6	1.00e+01	0.255	0.082	0.173
TQUARTIC	ARC-GLRT	63.9	282.1	64.9	52.9	–	2.37e-14	0.363	–	0.363
(1000)	ARC-RBB	71.5	838.6	72.5	55.6	6.9	1.99e-13	0.598	0.084	0.513
	ARC-RAPG	71.0	934.8	72.0	55.4	6.6	8.58e-11	0.674	0.090	0.584
	ARC-URBB	71.3	566.7	72.3	55.6	6.9	1.04e-10	0.521	0.091	0.431
	ARC-URAPG	72.3	926.4	73.3	56.5	6.7	3.79e-10	0.639	0.087	0.552
WOODS	ARC-GLRT	286.4	4 542.6	287.4	210.2	–	8.66e-15	1.561	–	1.561
(1000)	ARC-RBB	382.8	9 574.8	383.8	264.5	6.2	1.88e-12	3.733	0.067	3.666
	ARC-RAPG	381.3	9 426.5	382.3	263.9	5.5	3.15e-14	3.340	0.051	3.288
	ARC-URBB	382.2	9 486.3	383.2	264.2	6.1	1.67e-14	3.722	0.067	3.655
	ARC-URAPG	381.7	9 542.6	382.7	264.6	6.3	1.67e-12	3.833	0.068	3.765

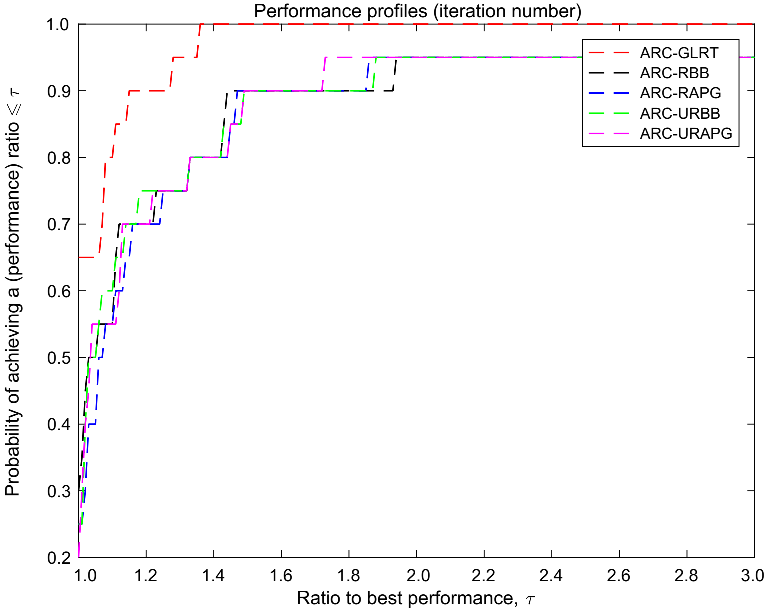

Fig. 1 Performance profiles for iteration number for ARC-GLRT, ARC-RBB, ARC-RAPG, ARC-URBB and ARC-URAPG on the CUTEst problems

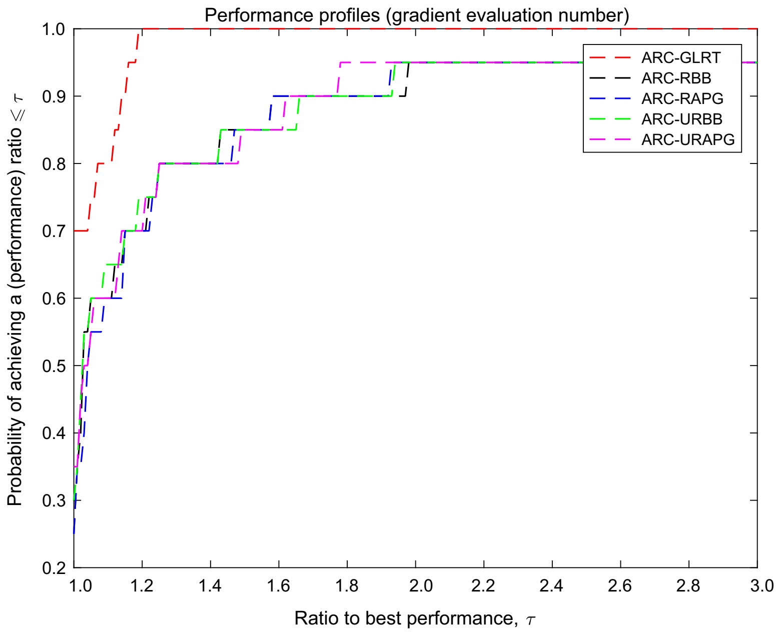

Fig. 2 Performance profiles for gradient evaluations for ARC-GLRT, ARC-RBB, ARC-RAPG, ARC-URBB and ARC-URAPG on the CUTEst problems

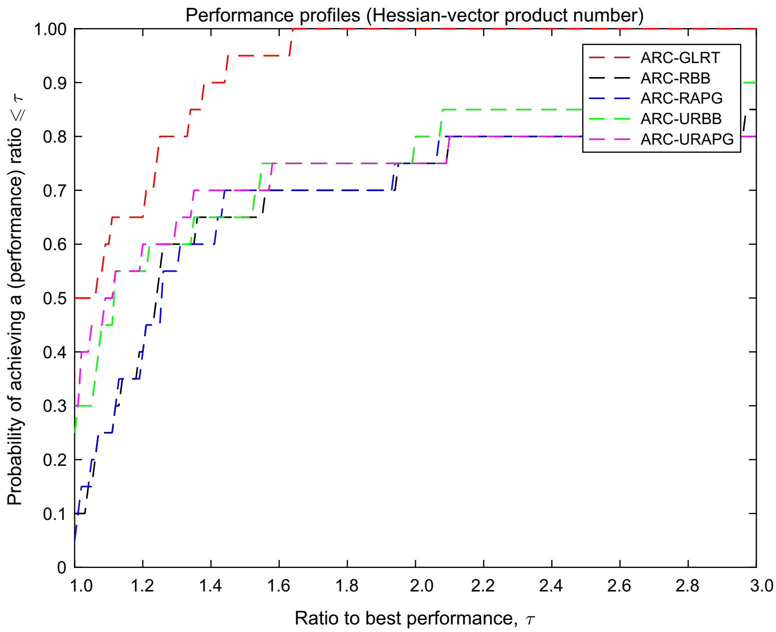

Fig. 3 Performance profiles for Hessian-vector products for ARC-GLRT, ARC-RBB, ARC-RAPG, ARC-URBB and ARC-URAPG on the CUTEst problems

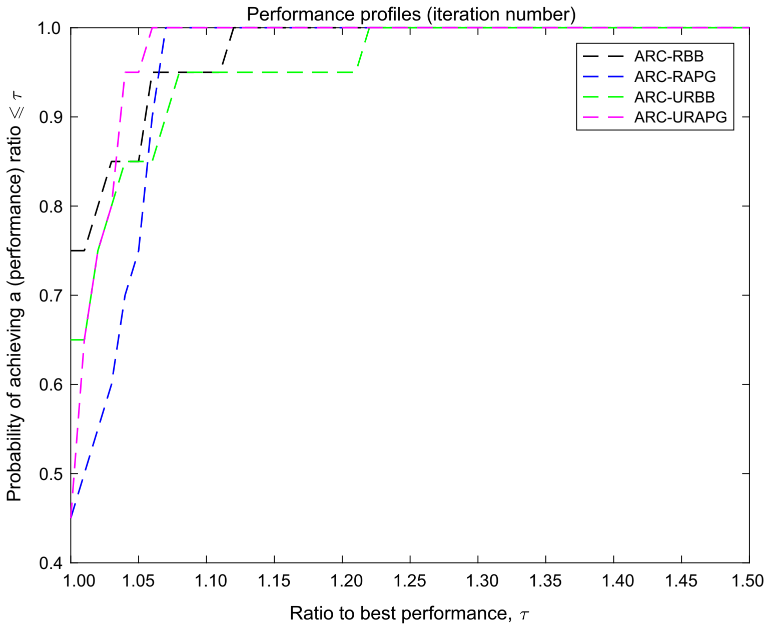

Fig. 4 Performance profiles for iteration number for ARC-RBB, ARC-RAPG, ARC-URBB and ARC-URAPG on the CUTEst problems

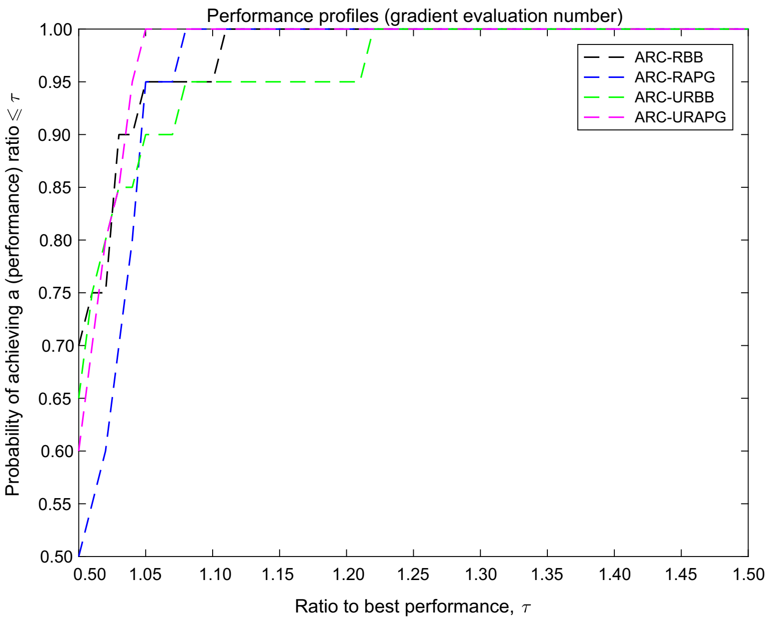

Fig. 5 Performance profiles for gradient evaluation for ARC-RBB, ARC-RAPG, ARC-URBB and ARC-URAPG on the CUTEst problems

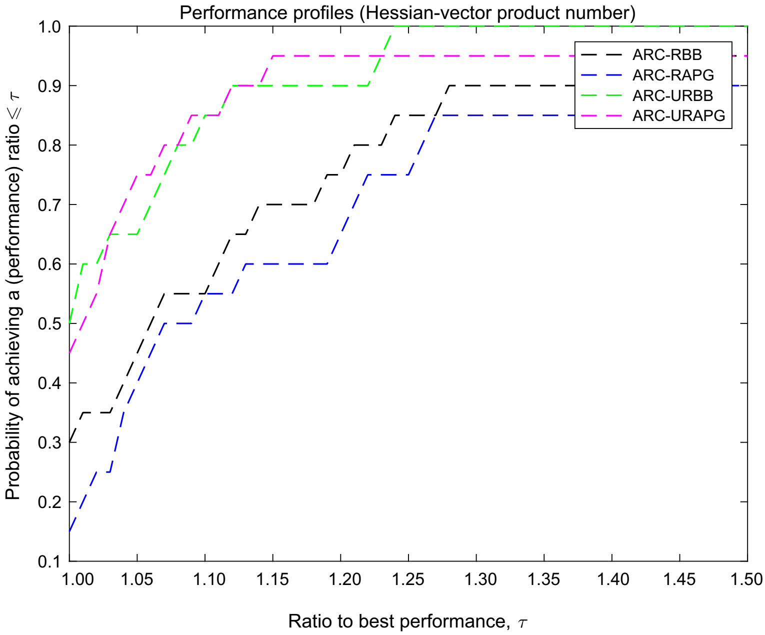

Fig. 6 Performance profiles for Hessian-vector products for `ARC-RBB`, `ARC-RAPG`, `ARC-URBB` and `ARC-URAPG` on the CUTEst problems

6 Conclusion

In this paper, we propose a new convex reformulation for the subproblem of the cubic regularization methods. Based on our reformulation, we propose a variant of the non-adaptive CR algorithm that admits an iteration complexity $\mathcal{O}(\epsilon_g^{-3/2})$ to find an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point. Moreover, we show that an operation complexity bound of our algorithm is $\tilde{\mathcal{O}}(\epsilon_g^{-7/4})$ when the subproblems are solved by Nesterov's accelerated gradient method and the approximated eigenvalues are computed by the Lanczos procedure. We also propose a variant of the ARC algorithm with similar complexity guarantees. Both of our iteration and operation complexity bounds match the best-known bounds in the literature for algorithms that based on first- and second-order information. Numerical experiments on the ARC equipped with our reformulation for solving subproblems also illustrate the effectiveness of our approach.

For future research, we would like to explore if our reformulation can be extended to solve auxiliary problems in tensor methods for unconstrained optimization ([27–30]), which were shown to have fast global convergence guarantees. It is well known that the auxiliary problem in the model function in each iteration of the tensor method is a regularized p-th-order Taylor approximation, which is difficult to solve. Two recent works [29, 30] show that for $p = 3$ and convex minimization problems, the Tensor model can be solved by an adaptive Bregman proximal gradient method, where each subproblem is of form

Table 2 The number of times ARC-URBB (or ARC-URAPG) performs better than the other three algorithms in 10 realizations.

Problem	Index	ARC-URBB			ARC-URAPG		
		ARC-GLRT	ARC-RBB	ARC-RAPG	ARC-GLRT	ARC-RBB	ARC-RAPG
BROYDN7D	n_i	4	5	2	4	6	4
	n_{prod}	4	5	6	5	8	6
	n_g	3	3	1	4	4	2
BRYBND	n_i	10	0	0	10	0	0
	n_{prod}	7	7	4	7	9	6
	n_g	10	0	0	10	0	0
CHAINWOO	n_i	0	5	7	0	3	5
	n_{prod}	0	10	9	0	10	9
	n_g	0	6	6	0	2	5
DIXMAANF	n_i	7	0	2	6	1	0
	n_{prod}	5	6	5	7	6	7
	n_g	6	0	1	6	2	0
DIXMAANG	n_i	7	5	4	6	6	3
	n_{prod}	10	10	9	9	10	9
	n_g	6	4	4	7	6	2
DIXMAANH	n_i	3	2	2	4	2	1
	n_{prod}	7	6	5	6	7	6
	n_g	4	2	2	3	2	1
DIXMAANJ	n_i	1	5	5	1	5	4
	n_{prod}	10	7	7	10	9	7
	n_g	1	6	5	1	3	3
DIXMAANK	n_i	0	7	7	1	5	4
	n_{prod}	9	7	9	8	8	7
	n_g	0	5	7	0	5	5
DIXMAANL	n_i	4	7	6	2	4	5
	n_{prod}	8	8	8	9	7	7
	n_g	2	6	6	1	3	5
EXTROSNB	n_i	10	3	3	10	2	3
	n_{prod}	0	3	3	0	2	3
	n_g	10	3	3	10	5	5
FLETCHCR	n_i	5	4	5	5	5	6
	n_{prod}	4	8	8	5	9	8
	n_g	2	2	6	2	2	9
FREUROTH	n_i	5	4	4	7	3	5
	n_{prod}	0	5	7	0	4	6
	n_g	5	5	6	4	4	7

Table 2 continued

Problem	Index	ARC-URBB			ARC-URAPG		
		ARC-GLRT	ARC-RBB	ARC-RAPG	ARC-GLRT	ARC-RBB	ARC-RAPG
GENHUMPS	n_i	10	1	0	10	2	0
	n_{prod}	10	7	8	10	7	7
	n_g	10	0	0	10	1	0
GENROSE	n_i	3	5	4	3	4	4
	n_{prod}	1	5	5	1	6	5
	n_g	3	4	6	3	5	5
NONCVXU2	n_i	0	7	6	0	9	7
	n_{prod}	6	4	6	7	8	6
	n_g	0	5	6	0	8	7
NONCVXUN	n_i	0	4	4	0	7	5
	n_{prod}	1	4	4	2	7	5
	n_g	0	4	4	0	7	5
OSCIPATH	n_i	0	4	3	0	4	5
	n_{prod}	5	4	3	5	4	5
	n_g	0	3	3	0	5	4
TOINTGSS	n_i	6	1	3	7	1	1
	n_{prod}	1	5	9	0	6	9
	n_g	7	2	4	7	2	1
TQUARTIC	n_i	4	3	4	3	4	4
	n_{prod}	0	9	10	0	4	7
	n_g	5	3	4	3	4	4
WOODS	n_i	0	6	1	0	4	4
	n_{prod}	0	6	4	0	6	3
	n_g	0	6	1	0	4	2

$$\min_x c^\top x + \frac{1}{2}x^\top Ax + \frac{\gamma}{3+\nu}\|x\|^{3+\nu}, \text{ with } A \succeq 0.$$

It will be interesting to see if the methods in [29, 30] can be extended to nonconvex minimization problems and still have similar subproblems, and if our reformulation can be extended to solving these subproblems.

Acknowledgements The first and third authors would like to thank Professor Duan Li, for his advice, help and encouragement during their Ph.D. and postdoctoral time in the Chinese University of Hong Kong. All the authors extend our thanks to the two anonymous referees for the invaluable comments that improve the quality of the paper significantly.

A Appendix

A.1 Proofs for Lemma 1

In this case, s_k is approximately computed by (7) or (9). Let $\sigma = L/2$ for Algorithm 1 and $\sigma = \sigma_k$ for Algorithm 2. Noting that $d_k = s_k$, by (13), we have

$$\|g_k + H_k d_k + 3\epsilon_E d_k + \sigma \|d_k\| d_k\| \leqslant \epsilon_S. \tag{A1}$$

Since

$$\|g_k + H_k d_k + 3\epsilon_E d_k + \sigma \|d_k\| d_k\| \geqslant \|g_k + H_k d_k\| - 3\epsilon_E \|d_k\| - \sigma \|d_k\|^2,$$

we have

$$\|g_k + H_k d_k\| \leqslant \epsilon_S + 3\epsilon_E \|d_k\| + \sigma \|d_k\|^2. \tag{A2}$$

Due to $\alpha_k \leqslant \lambda_{\min}(H_k) + \epsilon_E$ and $\alpha_k \geqslant -\epsilon_E$, we have

$$\lambda_{\min}(H_k) \geqslant -2\epsilon_E. \tag{A3}$$

Using (A1), we have

$$\begin{aligned} g_k^\top d_k + d_k^\top H_k d_k + 3\epsilon_E \|d_k\|^2 + \sigma \|d_k\|^3 &\leqslant \|g_k + H_k d_k + 3\epsilon_E d_k + \sigma \|d_k\| d_k\| \cdot \|d_k\| \\ &\leqslant \epsilon_S \|d_k\|. \end{aligned} \tag{A4}$$

Then, we have

$$\begin{aligned} g_k^\top d_k &\overset{(A4)}{\leqslant} \epsilon_S \|d_k\| - d_k^\top H_k d_k - 3\epsilon_E \|d_k\|^2 - \sigma \|d_k\|^3 \\ &= \epsilon_S \|d_k\| - (d_k^\top H_k d_k + 3\epsilon_E \|d_k\|^2) - \sigma \|d_k\|^3 \\ &\overset{(A3)}{\leqslant} \epsilon_S \|d_k\| - \epsilon_E \|d_k\|^2 - \sigma \|d_k\|^3. \end{aligned} \tag{A5}$$

By (A4), we also have

$$d_k^\top H_k d_k \leqslant -g_k^\top d_k - 3\epsilon_E \|d_k\|^2 - \sigma \|d_k\|^3 + \epsilon_S \|d_k\|. \tag{A6}$$

Hence, we obtain

$$
\begin{aligned}
m(d_k) &= g_k^\top d_k + \frac{1}{2} d_k^\top H_k d_k + \frac{\sigma}{3}\|d_k\|^3 \\
&\overset{(A6)}{\leqslant} g_k^\top d_k - \frac{1}{2} g_k^\top d_k - \frac{3}{2}\epsilon_E \|d_k\|^2 - \frac{\sigma}{2}\|d_k\|^3 + \frac{1}{2}\epsilon_S\|d_k\| + \frac{\sigma}{3}\|d_k\|^3 \\
&= \frac{1}{2} g_k^\top d_k - \frac{3}{2}\epsilon_E\|d_k\|^2 + \frac{1}{2}\epsilon_S\|d_k\| - \frac{\sigma}{6}\|d_k\|^3 \\
&\overset{(A5)}{\leqslant} \frac{1}{2}\epsilon_S\|d_k\| - \frac{\sigma}{2}\|d_k\|^3 - 2\epsilon_E\|d_k\|^2 + \frac{1}{2}\epsilon_S\|d_k\| - \frac{\sigma}{6}\|d_k\|^3 \\
&= -\frac{2\sigma}{3}\|d_k\|^3 - 2\epsilon_E\|d_k\|^2 + \epsilon_S\|d_k\|,
\end{aligned}
$$

or equivalently,

$$
-m(d_k) \geqslant \frac{2\sigma}{3}\|d_k\|^3 + 2\epsilon_E\|d_k\|^2 - \epsilon_S\|d_k\|.
$$

Due to $\epsilon_S = \epsilon_E^2/L$ as in Condition 2, it follows that

$$
-m(d_k) \geqslant \frac{2\sigma}{3}\|d_k\|^3 + 2\epsilon_E\|d_k\|^2 - \frac{\epsilon_E^2}{L}\|d_k\|. \tag{A7}
$$

Now suppose that $x_k + d_k$ is not an $(\epsilon_g, \sqrt{L\epsilon_g})$ stationary point. We then have either (i) $\|\nabla f(x_k+d_k)\| > \epsilon_g$ or (ii) $\|\nabla f(x_k+d_k)\| \leqslant \epsilon_g,\ \lambda_{\min}(\nabla^2 f(x_k+d_k)) < -\sqrt{L\epsilon_g}$. Let us consider the following cases (i) and (ii) separately.

(i) Using (A2), Taylor expansion for $\nabla f(x_k + d_k)$ and (11), we have

$$
\begin{aligned}
\|\nabla f(x_k + d_k)\| &\leqslant \|g_k + H_k d_k\| + \frac{L}{2}\|d_k\|^2 \\
&\leqslant \epsilon_S + 3\epsilon_E\|d_k\| + (\sigma + \frac{L}{2})\|d_k\|^2,
\end{aligned} \tag{A8}
$$

where the first inequality follows from Lemma 1 in [1]. Using $\|\nabla f(x_k+d_k)\| > \epsilon_g$ and (A8), we obtain

$$
\epsilon_g \leqslant \epsilon_S + 3\epsilon_E\|d_k\| + (\sigma + \frac{L}{2})\|d_k\|^2.
$$

This gives

$$
\|d_k\| \geqslant \frac{-3\epsilon_E + \sqrt{9\epsilon_E^2 - 4(\sigma + \frac{L}{2})(\epsilon_S - \epsilon_g)}}{2\sigma + L} \quad \text{or}
$$

$$
\|d_k\| \leqslant \frac{-3\epsilon_E - \sqrt{9\epsilon_E^2 - 4(\sigma + \frac{L}{2})(\epsilon_S - \epsilon_g)}}{2\sigma + L},
$$

where the second case is discarded since $\|d_k\| \geqslant 0$. Due to $\epsilon_S = \epsilon_E^2/L$ and $\epsilon_g = \epsilon_E^2/L$ in Condition 2, we further have

$$\|d_k\| \geqslant \frac{-3\epsilon_E + \sqrt{25\epsilon_E^2 + (32\sigma/L)\epsilon_E^2}}{2\sigma + L}. \tag{A9}$$

- For Algorithm 1, due to $\sigma = L/2$, (A9) yields

$$\|d_k\| \geqslant \frac{3}{2L}\epsilon_E.$$

  Therefore, using $\sigma = L/2$, we have

$$-m(d_k) \overset{(A7)}{\geqslant} \frac{33\epsilon_E^3}{8L^2}.$$

- Note that from Lemma 7, we have $\sigma = \sigma_k \leqslant \max\{\sigma_0, \frac{\gamma L}{2}\}$ for Algorithm 2. Therefore, if $\sigma \geqslant L/2$, (A9) yields

$$\|d_k\| \geqslant \frac{3}{(\max\{\gamma, 2\sigma_0/L\} + 1)L}\epsilon_E, \tag{A10}$$

  and if $\sigma < L/2$, (A9) yields

$$\|d_k\| \geqslant \frac{1}{L}\epsilon_E. \tag{A11}$$

  Using (A9), we claim the following inequality holds,

$$\frac{2\sigma}{3}\|d_k\|^2 + 2\epsilon_E\|d_k\| - \frac{\epsilon_E^2}{L} \geqslant \frac{3\epsilon_E^2}{L}, \tag{A12}$$

  provided $\sigma > 0$. Now combining (A7), (A10), (A11) and (A12), we have

$$-m(d_k) \geqslant \frac{3\epsilon_E^2}{L}\|d_k\| \geqslant \min\left\{\frac{3}{\max\{\gamma, 2\sigma_0/L\} + 1}, 1\right\} \cdot \frac{3\epsilon_E^3}{L^2}.$$

  Indeed, to prove inequality (A12), we only need to show, using (A9),

$$\frac{2(\sqrt{25 + 32a} - 3)^2}{3(4a + 4 + 1/a)} + \frac{2(\sqrt{25 + 32a} - 3)}{2a + 1} = \frac{2}{3}\frac{32a^2 + 16a + 3\sqrt{25 + 32a} - 9}{(2a + 1)^2} \geqslant 4,$$

  for $a = \sigma/L > 0$. Note that the above inequality is equivalent to

$$\psi(a) = 8a^2 - 8a + 3\sqrt{25 + 32a} - 15 \geqslant 0.$$

As $\psi(0) = 0$, it suffices to show

$$\psi'(a) = 16a - 8 + \frac{48}{\sqrt{25 + 32a}} > 0, \ \forall a \geqslant 0.$$

This holds because

$$\psi''(a) = 16 - 48\frac{16}{(\sqrt{25+32a})^3} = 16(1 - \frac{48}{(\sqrt{25+32a})^3}) \geqslant 16(1 - \frac{48}{125}) > 0, \ \forall a \geqslant 0,$$

and $\psi'(0) = \frac{8}{5} \geqslant 0$.

(ii) By Assumption 1 and (A3), we have

$$\lambda_{\min}(\nabla^2 f(x_k + d_k)) \geqslant \lambda_{\min}(H_k) - L\|d_k\| \geqslant -2\epsilon_E - L\|d_k\|. \tag{A13}$$

Using $\lambda_{\min}(\nabla^2 f(x_k + d_k)) \leqslant -\sqrt{L\epsilon_g}$ and (A13), we obtain

$$-2\epsilon_E - L\|d_k\| \leqslant -\sqrt{L\epsilon_g}.$$

This, together with $\epsilon_E = \sqrt{L\epsilon_g}/3$, implies

$$\|d_k\| \geqslant \frac{\sqrt{L\epsilon_g} - 2\epsilon_E}{L} = \frac{\epsilon_E}{L}. \tag{A14}$$

It follows that

$$\begin{aligned} -m(d_k) &\overset{(A7)}{\geqslant} \frac{2\sigma}{3}\|d_k\|^3 + 2\epsilon_E\|d_k\|^2 - \frac{\epsilon_E^2}{L}\|d_k\| \\ &\overset{(A14)}{\geqslant} \left(0 + 2\epsilon_E\left(\frac{\epsilon_E}{L}\right) - \frac{\epsilon_E^2}{L}\right)\|d_k\| \overset{(A14)}{\geqslant} \frac{\epsilon_E^3}{L^2}. \end{aligned}$$

Combining (i) and (ii), we complete the proof.

A.2 Proofs for Lemma 2

In this case, d_k is generated by either line 13 or line 16 of Algorithm 1 (Algorithm 2, respectively), depending on the norm of s_k returned by approximately solving (8) ((10), respectively). Let $\sigma = L/2$ for Algorithm 1 and $\sigma = \sigma_k$ for Algorithm 2. We prove the results twofold.

(i) When $\sigma\|s_k\| + \alpha_k \geqslant 0$, we must have $[\sigma\|s_k\| + \alpha_k]_+ = \sigma\|s_k\| + \alpha_k$. Note that

$$\nabla \tilde{m}_k^r(s_k) = g_k + H_k s_k + (2\epsilon_E - \alpha_k)s_k + [\sigma\|s_k\| + \alpha_k]_+ s_k. \tag{A15}$$

Using (14) and $d_k = s_k$, we obtain

$$\begin{aligned} & g_k^\top d_k + d_k^\top H_k d_k + (2\epsilon_E - \alpha_k)\|d_k\|^2 + [\sigma\|d_k\| + \alpha_k]_+ \|d_k\|^2 \\ \leqslant & \left\| g_k + H_k d_k + (2\epsilon_E - \alpha_k) d_k + [\sigma\|d_k\| + \alpha_k]_+ d_k \right\| \cdot \|d_k\| \\ \leqslant & \epsilon_S \|d_k\|. \end{aligned} \tag{A16}$$

Since $\alpha_k \leqslant \lambda_{\min}(H_k) + \epsilon_E$, we have $H_k - \alpha_k I + 2\epsilon_E I \succeq 0$ and thus

$$d_k^\top H_k d_k + (2\epsilon_E - \alpha_k)\|d_k\|^2 \geqslant 0. \tag{A17}$$

Noting that $[\sigma\|s_k\| + \alpha_k]_+ \geqslant 0$, we have from (A16) and (A17) that

$$g_k^\top d_k \leqslant \epsilon_S \|d_k\|. \tag{A18}$$

On the other hand, according to $\sigma\|s_k\| + \alpha_k \geqslant 0$ and (A16), we have

$$\begin{aligned} d_k^\top H_k d_k &\leqslant -g_k^\top d_k + (\alpha_k - 2\epsilon_E)\|d_k\|^2 - \sigma\|d_k\|^3 - \alpha_k\|d_k\|^2 + \epsilon_S\|d_k\| \\ &= -g_k^\top d_k - \sigma\|d_k\|^3 - 2\epsilon_E\|d_k\|^2 + \epsilon_S\|d_k\|. \end{aligned} \tag{A19}$$

Since $d_k = s_k$, and $\alpha_k \leqslant -\epsilon_E$, from $\sigma\|s_k\| + \alpha_k \geqslant 0$ we obtain

$$\|d_k\| \geqslant -\frac{\alpha_k}{\sigma} \geqslant \frac{\epsilon_E}{\sigma}. \tag{A20}$$

We further have

$$\begin{aligned} m(d_k) &= g_k^\top d_k + \frac{1}{2} d_k^\top H_k d_k + \frac{\sigma}{3}\|d_k\|^3 \\ &\overset{(A19)}{\leqslant} g_k^\top d_k - \frac{1}{2} g_k^\top d_k - \frac{\sigma}{2}\|d_k\|^3 - \epsilon_E\|d_k\|^2 + \frac{1}{2}\epsilon_S\|d_k\| + \frac{\sigma}{3}\|d_k\|^3 \\ &= \frac{1}{2} g_k^\top d_k + \frac{1}{2}\epsilon_S\|d_k\| - \epsilon_E\|d_k\|^2 - \frac{\sigma}{6}\|d_k\|^3 \\ &\overset{(A18)}{\leqslant} \frac{1}{2}\epsilon_S\|d_k\| + \frac{1}{2}\epsilon_S\|d_k\| - \epsilon_E\|d_k\|^2 - \frac{\sigma}{6}\|d_k\|^3 \\ &= -\frac{\sigma}{6}\|d_k\|^3 - \epsilon_E\|d_k\|^2 + \epsilon_S\|d_k\|. \end{aligned}$$

This gives

$$-m(d_k) \geqslant \frac{\sigma}{6}\|d_k\|^3 + \epsilon_E\|d_k\|^2 - \epsilon_S\|d_k\|. \tag{A21}$$

Thus, the desired bound holds immediately for Algorithm 1.
Now consider Algorithm 2. Similar to the proof for Lemma 1, we consider two cases $\|\nabla f(x_k + d_k)\| > \epsilon_g$, and $\|\nabla f(x_k + d_k)\| \leqslant \epsilon_g$, $\lambda_{\min}(\nabla^2 f(x_k + d_k)) < -\sqrt{L\epsilon_g}$. For the latter case, similar to case (ii) in the proof of Lemma 1, from

Lipschitz continuity of Hessian, we have $\|d_k\| \geqslant \epsilon_E/L$. Then, due to $\epsilon_S = \epsilon_E^2/L$ in Condition 2, we have $2\epsilon_E\|d_k\| - \epsilon_S \geqslant 0$ and thus (A21) yields

$$-m(d_k) \geqslant \frac{\sigma}{6}\|d_k\|^3 \overset{(A20)}{\geqslant} \frac{\epsilon_E^3}{6\sigma^2}.$$

Now consider the case $\|\nabla f(x_k + d_k)\| > \epsilon_g$, whose proof follows a similar idea to that in Lemma 1. Since the subproblem is approximately solved, we have

$$\|\nabla \tilde{m}_k^r(d_k)\| \leqslant \epsilon_S$$

and thus (A15), together with $\sigma\|d_k\| + \alpha_k \geqslant 0$, gives

$$\|g_k + H_k d_k + (2\epsilon_E - \alpha_k)d_k + [\sigma\|d_k\| + \alpha_k]_+ d_k\| = \|g_k + H_k d_k + (2\epsilon_E + \sigma\|d_k\|)d_k\| \leqslant \epsilon_S.$$

Hence, we have

$$\epsilon_g \leqslant \|\nabla f(x_k + d_k)\| \leqslant \|g_k + H_k\| + \frac{L}{2}\|d_k\|^2 \leqslant \epsilon_S + 2\epsilon_E\|d_k\| + (\sigma + \frac{L}{2})\|d_k\|^2.$$

The due to Condition 2, the above quadratic inequality gives

$$\|d_k\| \geqslant \frac{-2\epsilon_E + \sqrt{4\epsilon_E^2 - 4(\sigma + L/2)(\epsilon_S - \epsilon_g)}}{2\sigma + L} = \frac{-2\epsilon_E + \sqrt{20\epsilon_E^2 + 32\sigma\epsilon_E^2/L}}{2\sigma + L}.$$

We claim the following inequality holds

$$\begin{aligned} &\frac{\sigma}{6}\|d_k\|^2 + \epsilon_E\|d_k\| - \epsilon_S \\ &\quad \geqslant \frac{\sigma}{6}\left(\frac{-2\epsilon_E + \sqrt{20\epsilon_E^2 + 32\sigma\epsilon_E^2/L}}{2\sigma + L}\right)^2 \\ &\qquad + \epsilon_E\frac{-2\epsilon_E + \sqrt{20\epsilon_E^2 + 32\sigma\epsilon_E^2/L}}{2\sigma + L} - \frac{\epsilon_E^2}{L} \geqslant \frac{\epsilon_E^2}{3L}, \end{aligned} \tag{A22}$$

which further gives

$$-m(d_k) \geqslant \frac{1}{3L}\epsilon_E^2\|d_k\| \overset{(A20)}{\geqslant} \frac{1}{3\sigma L}\epsilon_E^3.$$

Indeed, (A22) is equivalent to, by defining $a = \sigma/L$,

$$\psi(a) = 4a\sqrt{5 + 8a} - 8a + 3\sqrt{5 + 8a} - 5 \geqslant 0,$$

which holds since

$$\frac{1}{8}\psi'(a) = \frac{6a+4}{\sqrt{5+8a}} - 1 \geqslant 0 \iff 36a^2+40a+9 \geqslant 0 \iff 9(2a+1)^2+4a \geqslant 0\ \forall\, a \geqslant 0,$$

and $\psi(0) = 3\sqrt{5} - 5 > 0$.

(ii) When $\sigma\|s_k\| + \alpha_k < 0$, we have $d_k = \frac{1}{2\sigma}w_k,\ \alpha_k = v_k^\top H_k v_k = \frac{w_k^\top H_k w_k}{w_k^\top w_k}$, $\|w_k\| = |\alpha_k|$, and $w_k^\top g_k \leqslant 0$. It follows that

$$d_k^\top H_k d_k = \frac{1}{4\sigma^2} w_k^\top H_k w_k = \frac{1}{4\sigma^2}\alpha_k w_k^\top w_k = \frac{1}{4\sigma^2}\alpha_k^3. \tag{A23}$$

Since $\alpha_k \leqslant -\epsilon_E < 0$, we also have

$$\|d_k\|^3 = \frac{1}{8\sigma^3}\|w_k\|^3 = \frac{1}{8\sigma^3}|\alpha_k|^3 = -\frac{1}{8\sigma^3}\alpha_k^3. \tag{A24}$$

Then, we have

$$\begin{aligned} m(d_k) &= g_k^\top d_k + \frac{1}{2}d_k^\top H_k d_k + \frac{\sigma}{3}\|d_k\|^3 \\ &= \frac{1}{2\sigma} g_k^\top w_k + \frac{1}{8\sigma^2}\alpha_k^3 - \frac{1}{24\sigma^2}\alpha_k^3 \\ &\leqslant \frac{1}{12\sigma^2}\alpha_k^3, \end{aligned}$$

where the second equality follows from (A23) and (A24), and the inequality follows from $w_k^\top g_k \leqslant 0$. Due to $\alpha_k \leqslant -\epsilon_E$, we have

$$-m(d_k) \geqslant -\frac{1}{12\sigma^2}\alpha_k^3 \geqslant \frac{1}{12\sigma^2}\epsilon_E^3.$$

Combining (i) and (ii) and noting $\sigma = L/2$ in Algorithm 1 and $\sigma = \sigma_k \in (0, \max\{\sigma_0, \gamma L/2\})$ (due to Lemma 7) and $\gamma > 1$ in Algorithm 2, we complete the proof.

References

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

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

[3] Griewank, A.: The modification of Newton's method for unconstrained optimization by bounding cubic terms. Technical report, Technical report NA/12, (1981)

[4] Cartis, C., Gould, N.I.M., Toint, P.L.: Adaptive cubic regularisation methods for unconstrained optimization. Part ii: worst-case function-and derivative-evaluation complexity. Math. Program. **130**(2), 295–319 (2011)

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

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

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

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

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

[10] Yair, C., Duchi, J.C.: Analysis of Krylov subspace solutions of regularized non-convex quadratic problems. In *Advances in Neural Information Processing Systems*, pp. 10705–10715 (2018)

[11] Carmon, Y., Duchi, J.C.: First-order methods for nonconvex quadratic minimization. SIAM Rev. **62**(2), 395–436 (2020)

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

[13] Jiang, Rujun, Yue, Man-Chung., Zhou, Zhishuo: An accelerated first-order method with complexity analysis for solving cubic regularization subproblems. Comput. Optim. Appl. **79**(2), 471–506 (2021)

[14] Flippo, O.E., Jansen, B.: Duality and sensitivity in nonconvex quadratic optimization over an ellipsoid. Eur. J. Oper. Res. **94**(1), 167–178 (1996)

[15] Ho-Nguyen, Nam, Kılınç-Karzan, Fatma: A second-order cone based approach for solving the trust-region subproblem and its variants. SIAM J. Optim. **27**(3), 1485–1512 (2017)

[16] Wang, Jiulin, Xia, Yong: A linear-time algorithm for the trust region subproblem based on hidden convexity. Optim. Lett. **11**(8), 1639–1646 (2017)

[17] Jiang, R., Li, D.: Novel reformulations and efficient algorithms for the generalized trust region subproblem. SIAM J. Optim. **29**(2), 1603–1633 (2019)

[18] Nesterov, Y.: Lectures on Convex Optimization, vol. 137. Springer, Berlin (2018)

[19] Xu, P., Roosta, F., Mahoney, M.W.: Newton-type methods for non-convex optimization under inexact Hessian information. Math. Program. **184**(1), 35–70 (2020)

[20] Vandenberghe, L..: *Accelerated Proximal Gradient Methods*. Lecture notes, https://www.seas.ucla.edu/~vandenbe/236C/lectures/fgrad.pdf, (2021)

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

[22] Jonathan, B., Borwein, J.M.: Two-point step size gradient methods. IMA J. Numer. Anal. **8**(1), 141–148 (1988)

[23] Gould, N.I.M., Orban, D., Toint, P.L.: Cutest: a constrained and unconstrained testing environment with safe threads for mathematical optimization. Comput. Optim. Appl. **60**(3), 545–557 (2015)

[24] O'Donoghue, Brendan, Candes, Emmanuel: Adaptive restart for accelerated gradient schemes. Found. Comput Math. **15**(3), 715–732 (2015)

[25] Ito, Naoki, Takeda, Akiko, Toh, Kim-Chuan.: A unified formulation and fast accelerated proximal gradient method for classification. J. Mach. Learn. Res. **18**(1), 510–558 (2017)

[26] Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002)

[27] Birgin, E.G., Gardenghi, J.L., Martínez, J.M., Santos, S.A., Toint, P.L.: Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models. Math. Program. **163**(1–2), 359–368 (2017)

[28] Jiang, Bo., Lin, Tianyi, Zhang, Shuzhong: A unified adaptive tensor approximation scheme to accelerate composite convex optimization. SIAM J. Optim. **30**(4), 2897–2926 (2020)

[29] Nesterov, Yurii: Implementable tensor methods in unconstrained convex optimization. Math. Program. **186**(1), 157–183 (2021)

[30] Geovani Nunes Grapiglia and Yu Nesterov: On inexact solution of auxiliary problems in tensor methods for convex optimization. Optim. Methods Softw. **36**(1), 145–170 (2021)