IMA Journal of Numerical Analysis (2023) **43**, 1855–1897
https://doi.org/10.1093/imanum/drac043
Advance Access publication on 22 August 2022

Inexact Newton-CG algorithms with complexity guarantees

ZHEWEI YAO
Department of Mathematics, University of California at Berkeley, Berkeley, CA 94720, USA

PENG XU
Amazon AWS AI (Work done while at the Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA 94305, USA)

FRED ROOSTA*
School of Mathematics and Physics, University of Queensland, Bisbane, QLD, 4072, Australia, The Centre for Information Resilience (CIRES), Brisbane, QLD, 4072, Australia, and International Computer Science Institute, Berkeley, CA, 94704, USA
*Corresponding author: fred.roosta@uq.edu.au

STEPHEN J. WRIGHT
Computer Sciences Department, University of Wisconsin-Madison, Madison, WI 53706, USA

AND

MICHAEL W. MAHONEY
International Computer Science Institute and Department of Statistics, University of California at Berkeley, Berkeley, CA 94704, USA

[Received on 27 September 2021; revised on 4 July 2022]

We consider variants of a recently developed Newton-CG algorithm for nonconvex problems (Royer, C. W. & Wright, S. J. (2018) Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization. *SIAM J. Optim.*, **28**, 1448–1477) in which inexact estimates of the gradient and the Hessian information are used for various steps. Under certain conditions on the inexactness measures, we derive iteration complexity bounds for achieving ϵ-approximate second-order optimality that match best-known lower bounds. Our inexactness condition on the gradient is adaptive, allowing for crude accuracy in regions with large gradients. We describe two variants of our approach, one in which the step size along the computed search direction is chosen adaptively, and another in which the step size is pre-defined. To obtain second-order optimality, our algorithms will make use of a negative curvature direction on some steps. These directions can be obtained, with high probability, using the randomized Lanczos algorithm. In this sense, all of our results hold with high probability over the run of the algorithm. We evaluate the performance of our proposed algorithms empirically on several machine learning models. Our approach is a first attempt to introduce inexact Hessian and/or gradient information into the Newton-CG algorithm of Royer & Wright (2018, Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization. *SIAM J. Optim.*, **28**, 1448–1477).

Keywords: Newton-CG; nonconvex optimization; inexact gradient; inexact Hessian.

1. Introduction

We consider the following unconstrained optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}), \tag{1.1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a smooth but nonconvex function. At the heart of many machine learning and scientific computing applications lies the problem of finding an (approximate) minimizer of (1.1). Faced with modern 'big data' problems, many classical optimization algorithms (Bertsekas, 1999; Nocedal & Wright, 2006) are inefficient in terms of memory and/or computational overhead. Much recent research has focused on approximating various aspects of these algorithms. For example, efficient variants of first-order algorithms, such as the stochastic gradient method, make use of inexact approximations of the gradient. The defining element of second-order algorithms is the use of the curvature information from the Hessian matrix. In these methods, the main computational bottleneck lies with evaluating the Hessian, or at least being able to perform matrix-vector products involving the Hessian. Evaluation of the gradient may continue to be an unacceptably expensive operation in second-order algorithms too. Hence, in adapting second-order algorithms to machine learning and scientific computing applications, we seek to *approximate* the computations involving the Hessian and the gradient, while preserving much of the convergence behavior of the exact underlying second-order algorithm.

Second-order methods use curvature information to nonuniformly rescale the gradient in a way that often makes it a more 'useful' search direction, in the sense of providing a greater decrease in function value. Second-order information also opens the possibility of convergence to points that satisfy second-order necessary conditions for optimality, that is, $\mathbf{x}$ for which $\|\nabla f(\mathbf{x})\| = 0$ and $\nabla^2 f(\mathbf{x}) \succeq \mathbf{0}$. For nonconvex machine learning problems, first-order stationary points include saddle points, which are undesirable for obtaining good generalization performance (LeCun *et al.*, 2012; Saxe *et al.*, 2013; Dauphin *et al.*, 2014; Choromanska *et al.*, 2015).

The canonical example of second-order methods is the classical Newton's method, which in its pure form is often written as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k, \quad \text{where } \mathbf{d}_k = -\mathbf{H}_k^{-1}\mathbf{g}_k,$$

where $\mathbf{H}_k = \nabla^2 f(\mathbf{x}_k)$ is the Hessian, $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$ is the gradient and α_k is some appropriate step size, often chosen using an Armijo-type line-search (Nocedal & Wright, 2006, Chapter 3). A more practical variant for large-scale problems is Newton-Conjugate-Gradient (Newton-CG), in which the linear system $\mathbf{H}_k\mathbf{d}_k = -\mathbf{g}_k$ is solved inexactly using the conjugate gradient (CG) algorithm (Steihaug, 1983). Such an approach requires access to the Hessian matrix only via matrix-vector products; it does not require $\mathbf{H}_k$ to be evaluated explicitly.

Recently, a new variant of the Newton-CG algorithm was proposed in Royer *et al.* (2020) that can be applied to large-scale nonconvex problems. This algorithm is equipped with certain safeguards and enhancements that allow worst-case complexity to be bounded in terms of the number of iterations and the total running time. However, this approach relies on the exact evaluation of the gradient and on matrix-vector multiplication involving the exact Hessian at each iteration. Such operations can be prohibitively expensive in machine learning problems. For example, when the underlying optimization

problem has the finite-sum form

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) = \sum_{i=1}^{n} f_i(\mathbf{x}), \tag{1.2}$$

exact computation of the Hessian/gradient can be costly when $n \gg 1$, requiring a complete pass through the training data set. Our work here builds upon that of Royer *et al.* (2020), but allows for inexactness in computation of gradients and Hessians, while obtaining a similar complexity result to the earlier paper.

1.1 *Related work*

Since deep learning became ubiquitous, first-order methods such as gradient descent and its adaptive, stochastic variants (Duchi *et al.*, 2011; Kingma & Ba, 2014), have become the most popular class of optimization algorithms in machine learning; see the recent textbooks (Beck, 2017; Lan, 2020; Lin *et al.*, 2020; Wright & Recht, 2021) for in-depth treatments. These methods are easy to implement, and their per-iteration cost is low compared to second-order alternatives. Although classical theory for first-order methods guarantees convergence only to first-order optimal (stationary) points, Ge *et al.* (2015); Levy (2016); Jin *et al.* (2017) argued that stochastic variants of certain first-order methods such as stochastic gradient descent (SGD) have the potential of escaping saddle points and converging to second-order stationary points. The effectiveness of such methods usually requires painstaking fine-tuning of their (often many) hyperparameters, and the number of iterations they require to escape saddle regions can be large.

By contrast, second-order methods can make use of curvature information (via the Hessian) to escape saddle points efficiently and ultimately converge to second-order stationary points. This behavior is seen in trust-region methods (Conn *et al.*, 2000; Curtis *et al.*, 2014, 2021), cubic regularization Nesterov & Polyak (2006) and its adaptive variants (ARC) (Cartis *et al.*, 2011a,b), as well as line-search based second-order methods (Royer & Wright, 2018; Royer *et al.*, 2020). Subsequent to Cartis *et al.* (2011a,b, 2012), which were among the first works to study Hessian approximations to ARC and trust region algorithms, respectively, Xu *et al.* (2020a) analyzed the optimal complexity of both trust region and cubic regularization, in which the Hessian matrix is approximated under milder conditions. Extension to gradient approximations was then studied in Tripuraneni *et al.* (2018); Yao *et al.* (2020). A novel take on inexact gradient and dynamic Hessian accuracy is investigated in Bellavia & Gurioli (2022). The analysis in Cartis & Scheinberg (2018); Gratton *et al.* (2018); Blanchet *et al.* (2019) relies on probabilistic models whose quality are ensured with a certain probability, but which allow for approximate evaluation of the objective function as well. Alternative approximations of the function and its derivative are considered in Bellavia *et al.* (2019).

A notable difficulty of these methods concerns the solution of their respective subproblems, which can themselves be nontrivial nonconvex optimization problems. Some exceptions are Roosta *et al.* (2018); Royer *et al.* (2020); Liu & Roosta (2021), whose fundamental operations are linear algebra computations, which are much better understood. While Roosta *et al.* (2018); Liu & Roosta (2021) are limited in their scope to invex problems (Mishra & Giorgi, 2008), the method in Royer *et al.* (2020) can be applied to more general nonconvex settings. In fact, Royer *et al.* (2020) enhances the classical Newton-CG approach with safeguards to detect negative curvature in the Hessian, during the solution of the Newton equations to obtain the step $\mathbf{d}_k$. Negative curvature directions can subsequently be exploited by the algorithm to make significant progress in reducing the objective. Moreover, Royer *et al.* (2020)

TABLE 1 *The upper bound of L_H for some nonconvex finite-sum minimization problems of the form (1.2). Here, we consider $\{(\mathbf{a}_i, b_i)\}_{i=1}^n$ as training data where $\mathbf{a}_i \in \mathbb{R}^d$ and $b_i \in \mathbb{R}$. For Welsch's exponential function ϕ, α is a positive parameter*

Problem formulation	Predictor function	Upper bound of L_H for single data point $(\mathbf{a}, b)$	Upper bound of L_H for entire problem
$\sum_{i=1}^n (b_i - \phi(\langle \mathbf{a}_i, \mathbf{x}\rangle))^2$	$\phi(z) = 1/(1+e^{-z})$	$2\|\mathbf{a}\|^3(\lvert b\phi'''(z)\rvert + 3\lvert\phi'(z)\phi''(z)\rvert + \lvert\phi(z)\phi'''(z)\rvert) \leqslant 2(\lvert b\rvert + 4)\|\mathbf{a}\|^3$	$\max_{i=1,\ldots,n}\ 2(\lvert b_i\rvert + 4)\|\mathbf{a}_i\|^3$
$\sum_{i=1}^n (b_i - \phi(\langle \mathbf{a}_i, \mathbf{x}\rangle))^2$	$\phi(z) = (e^z - e^{-z})/(e^z + e^{-z})$	$2\|\mathbf{a}\|^3(\lvert b\phi'''(z)\rvert + 3\lvert\phi'(z)\phi''(z)\rvert + \lvert\phi(z)\phi'''(z)\rvert) \leqslant 2(\lvert b\rvert + 4)\|\mathbf{a}\|^3$	$\max_{i=1,\ldots,n}\ 2(\lvert b_i\rvert + 4)\|\mathbf{a}_i\|^3$
$\sum_{i=1}^n \phi(b_i - \langle \mathbf{a}_i, \mathbf{x}\rangle)$	$\phi(z) = (1 - e^{-\alpha z^2})/\alpha$	$\|\mathbf{a}\|^3\lvert\phi'''(z)\rvert$	$9\alpha^{3/2}\max_{i=1,\ldots,n}\ \|\mathbf{a}_i\|^3$

TABLE 2 *The upper bound of K_g and K_H for the nonconvex finite-sum minimization problems of Table 1*

Problem formulation	Predictor function	Upper bound of K_g	Upper bound of K_H
$\sum_{i=1}^n (b_i - \phi(\langle \mathbf{a}_i, \mathbf{x}\rangle))^2$	$\phi(z) = 1/(1+e^{-z})$	$\max_{i=1,\ldots,n}\ (\lvert b_i\rvert + 1)\ \|\mathbf{a}_i\|/2$	$\max_{i=1,\ldots,n}\ (\lvert b_i\rvert + 2)\ \|\mathbf{a}_i\|^2$
$\sum_{i=1}^n (b_i - \phi(\langle \mathbf{a}_i, \mathbf{x}\rangle))^2$	$\phi(z) = (e^z - e^{-z})/(e^z + e^{-z})$	$\max_{i=1,\ldots,n}\ 2\left(\lvert b_i\rvert + 1\right)\ \|\mathbf{a}_i\|$	$\max_{i=1,\ldots,n}\ (\lvert b_i\rvert + 2)\ \|\mathbf{a}_i\|^2$
$\sum_{i=1}^n \phi(b_i - \langle \mathbf{a}_i, \mathbf{x}\rangle)$	$\phi(z) = (1 - e^{-\alpha z^2})/\alpha$	$\sqrt{2/\alpha}\max_{i=1,\ldots,n}\ \|\mathbf{a}_i\|$	$2\max_{i=1,\ldots,n}\ \|\mathbf{a}_i\|^2$

gives complexity guarantees that have been shown to be optimal in certain settings. (Henceforth, we use the term 'Newton-CG' to refer specifically to the algorithm in Royer *et al.*, 2020.)

1.2 *Contribution*

We describe two new variants of the Newton-CG algorithm of Royer *et al.* (2020) in which, to reduce overall computational costs, approximations of gradient and Hessian are employed. The first variant (Algorithm 3) is a line-search method in which only approximate gradient and Hessian information is needed at each step, but it resorts to the use of exact function values in performing a backtracking line search at each iteration. This requirement is not ideal, since exact evaluation of the objective function can be prohibitive. To partially remedy this situation, we propose a second variant (Algorithm 4), which, by employing constant step sizes, obviates the need for exact evaluations of functions, gradients or Hessians. The main drawback of this variant is that the fixed step size depends on bounds on problem-dependent quantities. While these are available in several problems of interest in machine learning and statistics (see Tables 1 and 2), they may be hard to estimate for other practical problems. Moreover, the step sizes obtained from these bounds tend to be conservative, a situation that arises often in fixed-step optimization methods.

For both of our proposed algorithms, we show that the convergence and complexity properties of the original exact algorithm from Royer *et al.* (2020) are largely retained. Specifically, to achieve $(\epsilon, \sqrt{\epsilon})$-optimality (see Definition 2.1 below) under Condition 2.2 on gradient and Hessian approximations (see below, in Section 2.3), we show the following.

- Inexact Newton-CG with backtracking line search (Algorithm 3) achieves the optimal iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$; see Section 2.3.

- Inexact Newton-CG in which a predefined step size replaces the backtracking line searches (Algorithm 4) achieves the same optimal iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$; see Section 2.4.
- We obtain estimates of oracle complexity in terms of ϵ for both variants.
- The accuracy required in our gradient approximation changes adaptively with the current gradient size. One consequence of this feature is to allow cruder gradient approximations in the regions with larger gradients, translating to a more efficient algorithm overall.
- We empirically illustrate the advantages of our methods on several real datasets; see Section 3.

We note that Algorithm 3 may not be computationally feasible as written, because the backtracking line searches require repeated (exact) evaluation of f. This requirement may not be practical in situations in which exact evaluations of f are impractical. By contrast, Algorithm 4 does not assume such knowledge and can be implemented strictly as written, given knowledge of the appropriate Lipschitz constant. The step-lengths used in Algorithm 4 are, however, quite conservative, and better computational results will almost certainly be obtained with Algorithm 3, modified to use approximations to $f(\mathbf{x})$; see the numerical examples in Section 3. This latter variant is not supported by our theory here. Nonetheless, our approach can be regarded as a first attempt to introduce inexact Hessian and/or gradient information into the Newton-CG algorithm of Royer *et al.* (2020).

2. Algorithms and analysis

We describe our algorithms and present our main theoretical results in this section. We start with background (Section 2.1) and important technical ingredients (Section 2.2), and then we proceed to our two main algorithms (Section 2.3 and Section 2.4).

2.1 *Notation, definitions and assumptions*

Throughout this paper, scalar constants are denoted by regular lower-case and upper-case letters, e.g., c and K. We use bold lowercase and blackboard bold uppercase letters to denote vectors and matrices, e.g., $\mathbf{a}$ and $\boldsymbol{A}$, respectively. The transpose of a real vector $\mathbf{a}$ is denoted by $\mathbf{a}^T$. For a vector $\mathbf{a}$, and a matrix $\boldsymbol{A}$, $\|\mathbf{a}\|$ and $\|\boldsymbol{A}\|$ denote the vector ℓ_2 norm and the matrix spectral norm, respectively. Subscripts (as in $\mathbf{a}_t$) denote iteration counters. The smallest eigenvalue of a symmetric matrix $\boldsymbol{A}$ is denoted by $\lambda_{\min}(\boldsymbol{A})$. For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, $[\mathbf{x}, \mathbf{y}]$ denotes the line segment between $\mathbf{x}$ and $\mathbf{y}$, i.e., $[\mathbf{x}, \mathbf{y}] = \{\mathbf{z} \mid \mathbf{z} = \mathbf{x} + \tau(\mathbf{y} - \mathbf{x}),\ 0 \leqslant \tau \leqslant 1\}$.

We are interested in expressing certain bounds in terms of their dependence on the small positive convergence tolerance ϵ, especially on certain negative powers of this quantity, ignoring the dependence on all other quantities in the problem, such as dimension, Lipschitz constants, etc. For example, we use $\mathcal{O}(\epsilon^{-1})$ to denote a bound that depends linearly on ϵ^{-1} and $\tilde{\mathcal{O}}(\epsilon^{-1})$ for linear dependence on $\epsilon^{-1}|\log\epsilon|$.

For nonconvex problems, the determination of near-optimality can be much more complicated than for convex problems; see the examples of Murty & Kabadi (1987); Hillar & Lim (2013). In this paper, as in earlier works (see for example Royer *et al.*, 2020), we make use of approximate second-order optimality, defined as follows.

DEFINITION 2.1 ((ϵ_g, ϵ_H)-optimality). Given $0 < \epsilon_g, \epsilon_H < 1$, $\mathbf{x}$ is an (ϵ_g, ϵ_H)-optimal solution of (1.1), if

$$\|\nabla f(\mathbf{x})\| \leqslant \epsilon_g \quad \text{and} \quad \lambda_{\min}(\nabla^2 f(\mathbf{x})) \geqslant -\epsilon_H. \tag{2.1}$$

ASSUMPTION 1 The smooth nonconvex function f is bounded below by the finite value f_{low}. It also has compact sub-level sets, i.e., the set $\mathscr{L}(\mathbf{x}_0) = \{\mathbf{x} \mid f(\mathbf{x}) \leqslant f(\mathbf{x}_0)\}$ is compact. Moreover, on an open set $\mathscr{B} \subset \mathbb{R}^n$ containing all line segments $[\mathbf{x}_k, \mathbf{x}_k + \mathbf{d}_k]$ for iterates $\mathbf{x}_k$ and search directions $\mathbf{d}_k$ generated by our algorithms, the objective function has Lipschitz continuous gradient and Hessian, that is, there are positive constants $0 < L_g < \infty$ and $0 < L_H < \infty$ such that for any $\mathbf{x}, \mathbf{y} \in \mathscr{B}$, we have

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leqslant L_g \|\mathbf{x} - \mathbf{y}\| \quad \text{and} \quad \left\|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{y})\right\| \leqslant L_H \|\mathbf{x} - \mathbf{y}\|.$$

Although Assumption 1 is typical in the optimization literature, it nonetheless implies a somewhat strong smoothness assumptions on the function. Some related works on various Newton-type methods, e.g., Bellavia *et al.* (2019); Bellavia & Gurioli (2022), obtain second-order complexity guarantees that require only Lipschitz continuity of the Hessian. It would be interesting to investigate whether our analysis can be modified to allow for such relaxations. We leave such investigations for future work.

Consequences of Lipschitz continuity of the Hessian, which we will use in later results, include the following bounds for any $\mathbf{x}, \mathbf{y} \in \mathscr{B}$:

$$\left\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y}) - \nabla^2 f(\mathbf{y})(\mathbf{x} - \mathbf{y})\right\| \leqslant \frac{L_H}{2}\|\mathbf{x} - \mathbf{y}\|^2 \tag{2.2a}$$

$$f(\mathbf{x}) \leqslant f(\mathbf{y}) + \nabla f(\mathbf{y})^T(\mathbf{x} - \mathbf{y}) + \frac{1}{2}(\mathbf{x} - \mathbf{y})^T \nabla^2 f(\mathbf{y})(\mathbf{x} - \mathbf{y}) + \frac{L_H}{6}\|\mathbf{x} - \mathbf{y}\|^3. \tag{2.2b}$$

An interesting avenue for future research is to try to replace these Lipschitz continuity conditions with milder variants in which the gradient and/or Hessian are required to maintain Lipschitz continuity only along a given set of directions, e.g., the piecewise linear path generated by the iterates such as the corresponding assumption in Xu *et al.* (2020a). Our current proof techniques do not allow for such relaxations, but we will look into possibility in future work.

For our inexact Newton-CG algorithms, we also require that the approximate gradient and Hessian satisfy the following conditions, for prescribed positive values $\delta_{g,t}$ and δ_H.

CONDITION 2.1 For given $\delta_{g,t}$ and δ_H, we say that the approximate gradient $\mathbf{g}_t$ and Hessian $\mathbf{H}_t$ at iteration t are $\delta_{g,t}$-accurate and δ_H-accurate if

$$\|\mathbf{g}_t - \nabla f(\mathbf{x}_t)\| \leqslant \delta_{g,t} \quad \text{and} \quad \|\mathbf{H}_t - \nabla^2 f(\mathbf{x}_t)\| \leqslant \delta_H,$$

respectively.

Under these assumptions and conditions, it is easy to show that there exist constants U_g and U_H such that the following are satisfied for all iterates $\mathbf{x}_t$ in the set defined in Assumption 1:

$$\|\mathbf{g}_t\| \leqslant U_g \quad \text{and} \quad \|\mathbf{H}_t\| \leqslant U_H. \tag{2.3}$$

2.2 *Key ingredients of the Newton-CG method*

We present the two major components from Royer *et al.* (2020) that are also used in our inexact variant of the Newton-CG algorithm. The first ingredient, Procedure 1 (referred to in some places as 'Capped CG'), is a version of the conjugate gradient (Shewchuk, 1994) algorithm that is used to solve a damped

Newton system of the form $\bar{\mathbf{H}}\mathbf{d} = -\mathbf{g}$, where $\bar{\mathbf{H}} = \mathbf{H} + 2\epsilon\mathbf{I}$ for some positive parameter ϵ. Procedure 1 is modified to detect indefiniteness in the matrix $\mathbf{H}$ and, when this occurs, to return a direction along which the curvature of $\mathbf{H}$ is at most $-\epsilon$. The second ingredient, Procedure 2 (referred to as the 'Minimum Eigenvalue Oracle' or 'MEO'), checks whether a direction of negative curvature (less than $-\epsilon$ for a given positive argument ϵ) exists for the given matrix $\mathbf{H}$. We now discuss each of these procedures in more detail.

Procedure 1 (Capped-CG). The well-known classical CG algorithm (Shewchuk, 1994) is used to solve linear systems involving positive definite matrices. However, this positive-definite requirement is often violated during the iterations for nonconvex optimization due to the indefiniteness of Hessians encountered at some iterates. Capped-CG, proposed by Royer *et al.* (2020) and presented in Procedure 1 for completeness, is an original way to leverage and detect such negative curvature directions, when they are encountered during CG iterations.

Lines 13–17 in Procedure 1 contain the standard CG operations. When $\mathbf{H} \succeq -\epsilon\mathbf{I}$, the tests in lines 22, 26 and 28 that indicate negative curvature will not be activated, and Capped-CG will return an approximate solution $\mathbf{d} \approx -\bar{\mathbf{H}}^{-1}\mathbf{g}$. However, when $\mathbf{H} \not\succeq -\epsilon\mathbf{I}$, Capped-CG will identify and return a direction of 'sufficient negative curvature'—a direction d satisfying $\mathbf{d}^T\mathbf{H}\mathbf{d} \leqslant -\epsilon\|\mathbf{d}\|^2$. Such a negative curvature direction is obtained under two circumstances. First, when the intermediate step (either $\mathbf{y}_j$ or $\mathbf{p}_j$) satisfies the negative curvature condition, that is, $\mathbf{d}^T\bar{\mathbf{H}}\mathbf{d} \leqslant -\epsilon\|\mathbf{d}\|^2$ (Lines 22 and 26), Procedure 1 will be terminated and the intermediate step will be returned. Secondly, when the residual, $\mathbf{r}_j$, decays at a slower rate than anticipated by standard CG analysis (Line 28), a negative curvature direction can be recovered by the procedure of Lines 29, 30 and 31. Note that Procedure 1 can be called with an optional input M, which is an upper bound on $\|\mathbf{H}\|$. However, even without *a priori* knowledge of this upper bound, M can be updated so that at any point in the execution of the procedure, M is an upper bound on the maximum curvature of $\mathbf{H}$ revealed to that point. Other parameters (κ, $\tilde{\zeta}$, τ, T) are also updated whenever the value of M changes. It is not hard to see that M is bounded by $U_{\mathbf{H}}$ throughout the execution of Procedure 1, provided that if an initial value of M is supplied to this procedure, this value satisfies $M \leqslant U_{\mathbf{H}}$.

Lemma 2.1 gives a bound on the number of iterations performed by Procedure 1.

LEMMA 2.1 (Royer *et al.*, 2020, Lemma 1). The number of iterations of Procedure 1 is bounded by

$$\min\{d, J(M, \epsilon, \zeta)\},$$

where $J = J(M, \epsilon, \zeta)$ is the smallest integer such that $\sqrt{T}(1-\tau)^{J/2} \leqslant \hat{\zeta}$. The number of matrix-vector products required is bounded by $2\min\{d, J(M, \epsilon, \zeta)\} + 1$, unless all iterates $\mathbf{y}_i$, $i = 1, 2, \ldots$ are stored, in which case it is $\min\{d, J(M, \epsilon, \zeta)\} + 1$. For the upper bound of $J(M, \epsilon, \zeta)$, we have

$$J(M, \epsilon, \zeta) \leqslant \min\left\{d, \tilde{\mathcal{O}}(\epsilon^{-1/2})\right\}. \tag{2.5}$$

When the slow decrease in residual is detected (Line 21), a direction of negative curvature for $\mathbf{H}$ can be extracted from the previous intermediate solutions, as the following result describes.

Procedure 1 Capped Conjugate Gradient

Inputs: Symmetric Matrix $\mathbf{H} \in \mathbb{R}^{d\times d}$, vector $\mathbf{g} \neq 0$; damping parameter $\epsilon \in (0,1)$; desired accuracy $\zeta \in (0,1)$;

Optional input: positive scale M (set to 0 if not provided)

Outputs $d_{\text{type}}, \mathbf{d}$

Secondary Output: $M, \kappa, \tilde{\zeta}, \tau, T$

Set

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

$\mathbf{y}_0 \leftarrow 0, \mathbf{r}_0 \leftarrow \mathbf{g}, \mathbf{p}_0 \leftarrow -\mathbf{g}, j \leftarrow 0$

if $\mathbf{p}_0^T \bar{\mathbf{H}} \mathbf{p}_0 < \epsilon \left\|\mathbf{p}_0\right\|^2$ **then**

Set $\mathbf{d} = p_0$ and terminate with $d_{\text{type}} = \text{NC}$;

else if $\|\mathbf{H}\mathbf{p}_0\| > M\|\mathbf{p}_0\|$ **then**

$M \leftarrow \|\mathbf{H}\mathbf{p}_0\|/\|\mathbf{p}_0\|$ and update $\kappa, \tilde{\zeta}, \tau, T$;

end if

while TRUE **do**

  $\alpha_j \leftarrow \mathbf{r}_j^T \mathbf{r}_j / \mathbf{p}_j^T \bar{\mathbf{H}} \mathbf{p}_j$; (Traditional CG Begins)

  $\mathbf{y}_{j+1} \leftarrow \mathbf{y}_j + \alpha_j \mathbf{p}_i$;

  $\mathbf{r}_{j+1} \leftarrow \mathbf{r}_j + \alpha_j \bar{\mathbf{H}} \mathbf{p}_j$;

  $\beta_{j+1} \leftarrow \mathbf{r}_{j+1}^T \mathbf{r}_{j+1} / \mathbf{r}_j^T \mathbf{r}_j$;

  $\mathbf{p}_{j+1} \leftarrow -\mathbf{r}_{j+1} + \beta_{j+1} \mathbf{p}_j$; (Traditional CG Ends)

  $j \leftarrow j+1$;

  if $\max(\|\mathbf{H}\mathbf{p}_j\|/\|\mathbf{p}_j\|, \|\mathbf{H}\mathbf{y}_j\|/\|\mathbf{y}_j\|, \|\mathbf{H}\mathbf{r}_j\|/\|\mathbf{r}_j\|) > M$ **then**

    $M \leftarrow \max(\|\mathbf{H}\mathbf{p}_j\|/\|\mathbf{p}_j\|, \|\mathbf{H}\mathbf{y}_j\|/\|\mathbf{y}_j\|, \|\mathbf{H}\mathbf{r}_j\|/\|\mathbf{r}_j\|)$ and update $\kappa, \tilde{\zeta}, \tau, T$;

  end if

  if $\mathbf{y}_j^T \bar{\mathbf{H}} \mathbf{y}_j \leqslant \epsilon \left\|\mathbf{y}_j\right\|^2$ **then**

    Set $\mathbf{d} \leftarrow \mathbf{y}_j$ and terminate with $d_{\text{type}} = \text{NC}$;

  else if $\left\|\mathbf{r}_j\right\| \leqslant \hat{\zeta} \left\|\mathbf{r}_0\right\|$ **then**

Continued

Set $\mathbf{d} \leftarrow \mathbf{y}_j$ and terminate with $d_{\text{type}} = \text{SOL}$;

else if $\mathbf{p}_j^T \bar{\mathbf{H}} \mathbf{p}_j \leqslant \epsilon \left\| \mathbf{p}_j \right\|^2$ **then**

Set $\mathbf{d} \leftarrow \mathbf{p}_j$ and terminate with $d_{\text{type}} = \text{NC}$;

else if $\left\| \mathbf{r}_j \right\| \geqslant \sqrt{T}(1-\tau)^{j/2} \left\| \mathbf{r}_0 \right\|$ **then**

Compute $\alpha_j, \mathbf{p}_{j+1}$ as in the main loop above;

Find $i \in \{0, \cdots, j-1\}$ such that

$$\frac{(\mathbf{y}_{j+1} - \mathbf{y}_i)^T \bar{\mathbf{H}} (\mathbf{y}_{j+1} - \mathbf{y}_i)}{\left\| \mathbf{y}_{j+1} - \mathbf{y}_i \right\|^2} \leqslant \epsilon; \tag{2.4}$$

Set $\mathbf{d} \leftarrow \mathbf{y}_{j+1} - \mathbf{y}_i$ and terminate with $d_{\text{type}} = \text{NC}$;

end if

end while

Return: d

LEMMA 2.2 (Royer *et al.*, 2020, Theorem 2). Suppose that the loop of Procedure 1 terminates with $j = \hat{J}$, where

$$\hat{J} \in \{1, 2, \ldots, \min\{n, J(M, \epsilon, \zeta)\}\}$$

satisfies

$$\|r_{\hat{J}}\| > \max\{\hat{\zeta}, \sqrt{T}(1-\tau)^{\hat{J}/2}\} \|r_0\|.$$

Suppose further that $y_{\hat{J}}^T \bar{\mathbf{H}} y_{\hat{J}} \geqslant \epsilon \|y_{\hat{J}}\|^2$, so that $y_{\hat{J}+1}$ is computed. Then we have

$$\frac{(y_{\hat{J}+1} - y_i)^T \bar{\mathbf{H}} (y_{\hat{J}+1} - y_i)}{\|y_{\hat{J}+1} - y_i\|^2} < \epsilon, \quad \text{for some } i \in \{0, \ldots, \hat{J}-1\}.$$

Note that $d^T \bar{\mathbf{H}} d \leqslant \epsilon \|d\|^2 \iff d^T \mathbf{H} d \leqslant -\epsilon \|d\|^2$.

Procedure 1 is invoked by the Newton-CG procedure, Algorithm 3 (described in Section 2.3), when the current iterate $\mathbf{x}_k$ has $\|\mathbf{g}_k\| \geqslant \epsilon_g > 0$. Procedure 1 can either return the approximate Newton direction or a negative curvature one. After describing how this output vector is modified by Algorithm 3, in the next section, we state a result (Lemma 2.4) about the properties of the resulting step.

In the case of $\|\mathbf{g}_k\| < \epsilon_g$, Algorithm 3 calls Procedure 2 to explicitly seek a direction of sufficient negative curvature. We describe this procedure next.

Procedure 2 Minimum Eigenvalue Oracle

1: **Inputs:** Symmetric matrix $\mathbf{H} \in \mathbb{R}^{d\times d}$, scalar $M \geqslant \lambda_{\max}(\mathbf{H})$ and $\epsilon > 0$;

2: Set $\delta \in [0, 1)$;

3: **Outputs:** Estimate λ of $\lambda_{\min}(\mathbf{H})$ such that $\lambda \leqslant -\epsilon/2$ and vector $\mathbf{v}$ with $\|\mathbf{v}\| = 1$ such that $\mathbf{v}^T\mathbf{H}\mathbf{v} = \lambda$ OR certificate that $\lambda_{\min}(\mathbf{H}) \geqslant -\epsilon$. The probability that the certificate is issued, but $\lambda_{\min}(\mathbf{H}) < -\epsilon$ is at most δ.

Procedure 2 (Minimum Eigenvalue Oracle). This procedure searches for a direction spanned by the negative spectrum of a given symmetric matrix or, alternately, verifies that the matrix is (almost) positive definite. Specifically, for a given $\epsilon > 0$, Procedure 2 finds a negative curvature direction $\mathbf{v}$ of $\mathbf{H}_k$ such that $\mathbf{v}^T\mathbf{H}\mathbf{v} \leqslant -\epsilon\|\mathbf{v}\|^2/2$, or else certifies that $\mathbf{H} \succeq -\epsilon\mathbf{I}$. The probability that the certificate is issued, but $\lambda_{\min}(\mathbf{H}) < -\epsilon$ is bounded above by some (small) specified value δ. As indicated in Royer *et al.* (2020), this minimum eigenvalue oracle can be implemented using the Lanczos process or the classical CG algorithm. (In this paper, we choose the former.) Both of these approaches have the same complexity, given in the following result.

LEMMA 2.3 (Royer *et al.*, 2020, Lemma 2). Suppose that the Lanczos method is used to estimate the smallest eigenvalue of $\mathbf{H}$ starting from a random vector drawn from the uniform distribution on the unit sphere, where $\|\mathbf{H}\| \leqslant M$. For any $\delta \in (0, 1)$, this approach finds the smallest eigenvalue of H to an absolute precision of $\epsilon/2$, together with a corresponding direction $\mathbf{v}$, in at most

$$\min\left\{d, 1 + \left\lceil \frac{\ln(2.75d/\delta^2)}{2}\sqrt{\frac{M}{\epsilon}}\right\rceil\right\} \quad \text{iterations}, \tag{2.6}$$

with probability at least $1 - \delta$. Each iteration requires evaluation of a matrix-vector product involving $\mathbf{H}$.

2.3 *Inexact Newton-CG algorithm with line search*

Algorithm 3 shows our inexact damped Newton-CG algorithm, which calls Procedures 1 and 2. In this section, we establish worst case iteration complexity to achieve (ϵ_g, ϵ_H)-optimality according to Definition 2.1. Under mild conditions on the approximate gradient and Hessian, the complexity estimate is the same as for the exact Newton-CG algorithm described in Royer *et al.* (2020).

For Algorithm 3, approximations of the Hessian and gradient can be used throughout. However, to obtain the step-size α_k, Algorithm 3 requires exact evaluation of the function. We avoid the need for these exact evaluations in the fixed-step variant, Algorithm 4, to be studied in Section 2.4.

Apart from the use of approximate Hessian and gradient, Lines 9–18 constitute a notable difference between our algorithm and the exact counterpart of Royer *et al.* (2020), in which our method calls Procedure 2 to obtain a direction of sufficient negative curvature when the direction $\mathbf{d}_k$ derived from Procedure 1 is small; specifically, $\|\mathbf{d}_k\| \leqslant \epsilon_g/\epsilon_H$. If such a direction is found, we perform a backtracking line search along with it. Otherwise, if Procedure 2 certifies that no direction of sufficient negative

Algorithm 3 Inexact Damped Newton-CG with Line Search

Inputs: $\epsilon_g, \epsilon_H > 0$; backtracking parameter $\theta \in (0,1)$; sufficient decrease parameter $\eta > 0$; starting point $\mathbf{x}_0$; upper bound on Hessian norm $U_H > 0$; accuracy parameter $\zeta \in (0, \min\{1, U_H\})$;
for $k = 0, 1, 2, \cdots$ **do**
  if $\|\mathbf{g}_k\| \geqslant \epsilon_g$ **then**
    Call Procedure 1 with $\mathbf{H} = \mathbf{H}_k, M = U_H, \epsilon = \epsilon_H, \mathbf{g} = \mathbf{g}_k$ and accuracy parameter ζ to obtain $\mathbf{d}$ and d_{type};
  if $d_{\text{type}} ==$ NC **then**
    $\mathbf{d}_k \leftarrow -\text{sgn}(\mathbf{d}^T\mathbf{g}_k)\frac{|\mathbf{d}^T\mathbf{H}_k\mathbf{d}|}{\|\mathbf{d}\|^2}\frac{\mathbf{d}}{\|\mathbf{d}\|}$ and go to **Line-Search:**
  else
    $\mathbf{d}_k \leftarrow \mathbf{d}$;
    if $\|\mathbf{d}_k\| \leqslant \epsilon_g/\epsilon_H$ **then**
    Call Procedure 2 with $\mathbf{H} = \mathbf{H}_k, M = U_H, \epsilon = \epsilon_H$ to obtain $\mathbf{v}$ (with $\|\mathbf{v}\| = 1$ and $\mathbf{v}^T\mathbf{H}_k\mathbf{v} \leqslant -\epsilon_H/2$) or a certificate that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon_H$;
    if Procedure 2 certifies that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon_H$ **then**
      Terminate and return $\mathbf{x}_k + \mathbf{d}_k$;
    else
      $\mathbf{d}_k \leftarrow -\left(\text{sgn}(\mathbf{v}^T\mathbf{g}_k)|\mathbf{v}^T\mathbf{H}_k\mathbf{v}|\right)\mathbf{v}$, $d_{\text{type}} \leftarrow$ NC, and go to **Line-Search**;
    end if
  else
    Go to **Line-Search;**
  end if
 end if
else
  $d_{\text{type}} \leftarrow$ NC;
  Call Procedure 2 with $\mathbf{H} = \mathbf{H}_k, M = U_H, \epsilon = \epsilon_H$ to obtain $\mathbf{v}$ with $\|\mathbf{v}\| = 1$ and $\mathbf{v}^T\mathbf{H}_k\mathbf{v} \leqslant -\epsilon_H/2$ or a certificate that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon_H$;
  if Procedure 2 certifies that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon_H$
    Terminate and return $\mathbf{x}_k$;
  else
    $\mathbf{d}_k \leftarrow -\text{sgn}(\mathbf{v}^T\mathbf{g}_k)|\mathbf{v}^T\mathbf{H}_k\mathbf{v}|\mathbf{v}$ and go to **Line-Search;**
  end if

Continued

28: **end if**

29: **Line-Search:**

30: **if** $d_{\text{type}} ==$ SOL **then**

31: Set $\alpha_k \leftarrow \theta^{j_k}$, where j_k is the smallest nonnegative integer such that

$$f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) < f(\mathbf{x}_k) - \frac{\eta}{6}|\alpha_k|^3 \|\mathbf{d}_k\|^3; \tag{2.7}$$

32: **else**

33: Set α_k to be the first element of the sequence $1, -1, \theta, -\theta, \theta^2, -\theta^2, \theta^3, -\theta^3, \ldots$ for which (2.7) holds;

34: **end if**

35: $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \alpha_k \mathbf{d}_k$;

36: **end for**

curvature exists, we terminate and return the point $\mathbf{x}_k + \mathbf{d}_k$, which already satisfies the second-order optimality condition. In theory, this modification is critical to obtaining the optimal worst-case complexity. In practice, however, we have observed that performing line-search with such $\mathbf{d}_k$, despite the fact that $\|\mathbf{d}_k\| \leqslant \epsilon_g/\epsilon_H$, results in acceptable progress in reducing the function. In other words, we believe that Lines 9-16 of Algorithms 3 and 4 serve a mainly theoretical purpose, and can be safely omitted in practical implementations.

Another notable difference with previous versions of this general approach is the use of a 'bidirectional' line search when $\mathbf{d}_k$ is a negative curvature direction. We do backtracking along both positive and negative directions, $\mathbf{d}_k$ and $-\mathbf{d}_k$, because we are unable to determine with certainty the sign of $\mathbf{d}_k^T \nabla f(\mathbf{x}_k)$, since we have access only to the approximation $\mathbf{g}_k$ of $\nabla f(x_k)$. This additional algorithmic feature causes only modest changes to the analysis of the function decrease along negative curvature directions, as we point out in the appropriate results below.

We begin our complexity analysis with a result that summarizes important properties of the direction $\mathbf{d}_k$ that is derived from the capped CG algorithm, Procedure 1. (The proof is identical to that of the cited result, Royer *et al.*, 2020, Lemma 3, except that we use approximate values of the Hessian and gradient of f here.)

LEMMA 2.4 (Royer *et al.*, 2020, Lemma 3). Suppose that Assumption 1 is satisfied. Suppose that Procedure 1 is invoked at an iterate $\mathbf{x}_k$ of Algorithm 3 (so that $\|\mathbf{g}_k\| \geqslant \epsilon_g > 0$) with inputs $\mathbf{H} = \mathbf{H}_k$, $\mathbf{g} = \mathbf{g}_k$, $\epsilon = \epsilon_H$ and ζ. Suppose that $\mathbf{d}_k$ in Algorithm 3 is obtained from the output vector $\mathbf{d}$ of Procedure 1, after possible scaling and change of sign. Then one of the two following statements holds.

1. $d_{\text{type}} =$ SOL and $\mathbf{d}_k = \mathbf{d}$ satisfies

$$\mathbf{d}_k^T \mathbf{H}_k \mathbf{d}_k \geqslant -\epsilon_H \|\mathbf{d}_k\|^2, \tag{2.8a}$$

$$\|\mathbf{d}_k\| \leqslant 1.1\epsilon_H^{-1}\|\mathbf{g}_k\|, \tag{2.8b}$$

$$\|\hat{\mathbf{r}}_k\| \leqslant \frac{1}{2}\epsilon_H\zeta\|\mathbf{d}_k\|, \tag{2.8c}$$

where

$$\hat{\mathbf{r}}_k = (\mathbf{H}_k + 2\epsilon_H\mathbf{I})\mathbf{d}_k + \mathbf{g}_k. \tag{2.9}$$

2. $d_{\text{type}} = \text{NC}$ and $\mathbf{d}_k$ satisfies

$$\mathbf{d}_k = -\text{sgn}(\mathbf{d}^T\mathbf{g}_k)\frac{|\mathbf{d}^T\mathbf{H}_k\mathbf{d}|}{\|\mathbf{d}\|^2}\frac{\mathbf{d}}{\|\mathbf{d}\|},$$

and $\mathbf{d}_k$ satisfies

$$\frac{\mathbf{d}_k^T\mathbf{H}_k\mathbf{d}_k}{\|\mathbf{d}_k\|^2} = -\|\mathbf{d}_k\| \leqslant -\epsilon_H. \tag{2.10}$$

In order to establish the iteration complexity of Algorithm 3, we first present a sufficient condition on the degree of the inexactness of the gradient and Hessian.

CONDITION 2.2 We require the inexact gradient $\mathbf{g}_k$ and Hessian $\mathbf{H}_k$ to satisfy Condition 2.1 with

$$\delta_{g,k} \leqslant \frac{1-\zeta}{8}\max\left(\epsilon_g, \min\left(\epsilon_H\|\mathbf{d}_k\|, \|\mathbf{g}_k\|, \|\mathbf{g}_{k+1}\|\right)\right) \quad \text{and} \quad \delta_H \leqslant \left(\frac{1-\zeta}{4}\right)\epsilon_H.$$

One could simplify Condition 2.2 to have an iteration-independent condition on $\delta_{g,k} \equiv \delta_g$, namely,

$$\delta_g \leqslant \frac{1-\zeta}{8}\epsilon_g.$$

However, the adaptivity of the iteration-dependent version of Condition 2.2 through $\mathbf{g}_k$ and $\mathbf{g}_{k+1}$ offers practical advantages. Indeed, in many iterations, one can expect $\|\mathbf{g}_k\|$ and $\|\mathbf{g}_{k+1}\|$ to be of similar magnitudes. Also, as shown in Lemma 2.4, we have $\|\mathbf{d}_k\| \leqslant 1.1\epsilon_H^{-1}\|\mathbf{g}_k\|$. Thus, the three terms in $\min(\epsilon_H\|\mathbf{d}_k\|, \|\mathbf{g}_k\|, \|\mathbf{g}_{k+1}\|)$ are often roughly of the same order, and usually larger than ϵ_g. These observations suggest that when the true gradient is large, we can employ loose approximations.

Given Condition 2.2, the proofs of the complexity bounds boil down to three parts. First, we bound the decrease in the objective function $f(\mathbf{x}_k)$ (Lemma 2.5) when taking the damped Newton step $\mathbf{d}_k$ (that is, when $d_{\text{type}} = \text{SOL}$ on return from Procedure 1 and $\|\mathbf{d}_k\|$ is not too small). Secondly, we bound the decrease in the objective when a negative curvature direction is encountered in Procedure 1 (Lemma 2.6) or Procedure 2 (Lemma 2.7). Thirdly, for Lines 9–18 in Algorithm 3, we show that the algorithm can be terminated after the update in Line 12. In particular, when the update direction is sufficiently small from Procedure 1 and a large negative curvature from Procedure 2 has not been detected, Line 12 terminates at a point satisfying the required optimality conditions (Lemma 2.8).

We start with the case in which an inexact Newton step is used.

LEMMA 2.5 Suppose that Assumption 1 is satisfied and that Condition 2.2 holds for all k. Suppose that at iteration k of Algorithm 3, we have $\|\mathbf{g}_k\| \geqslant \epsilon_g$, so that Procedure 1 is called. When Procedure 1 outputs a direction $\mathbf{d}_k$ with $d_{\text{type}} = \text{SOL}$ and $\|\mathbf{d}_k\| > \epsilon_g/\epsilon_H$, then the backtracking line search requires at most $j_k \leqslant j_{\text{sol}} + 1$ iterations, where

$$j_{\text{sol}} = \left\lceil \frac{1}{2} \log_\theta \left(\frac{3(1-\zeta)\epsilon_H^2}{4.4 U_g (L_H + \eta)} \right) \right\rceil,$$

and the resulting step $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$ satisfies

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant c_{\text{sol}} \max \left\{ 0, \min \left(\frac{(\|\mathbf{g}_{k+1}\| - \delta_{g,k} - \delta_{g,k+1})^3}{(2.5\epsilon_H)^3}, (2.5\epsilon_H)^3, \epsilon_g^{3/2} \right) \right\}, \tag{2.11}$$

where

$$c_{\text{sol}} = \frac{\eta}{6} \min \left\{ \frac{1}{(1+2L_H)^{3/2}}, \left[\frac{3\theta^2(1-\zeta)}{4(L_H+\eta)} \right]^{3/2} \right\}.$$

Proof. When the $d_{\text{type}} = \text{SOL}$, $\mathbf{d}_k$ is the solution of the inexact regularized Newton equations. We first prove that when $\mathbf{d}_k^T \mathbf{g}_k < 0$, the inner product $\mathbf{d}_k^T \nabla f(\mathbf{x}_k)$ is also negative:

$$\begin{aligned}
\mathbf{d}_k^T \nabla f(\mathbf{x}_k) &\leqslant \mathbf{d}_k^T \mathbf{g}_k + \delta_{g,k} \|\mathbf{d}_k\| \\
&= \mathbf{d}_k^T \hat{\mathbf{r}}_k - \mathbf{d}_k^T (\mathbf{H}_k + 2\epsilon_H \mathbf{I}) \mathbf{d}_k + \delta_{g,k} \|\mathbf{d}_k\| && \text{(from (2.9))} \\
&\leqslant \|\mathbf{d}_k\| \|\hat{r}_k\| - \epsilon_H \|\mathbf{d}_k\|^2 + \delta_{g,k} \|\mathbf{d}_k\| && \text{(from (2.8a))} \\
&\leqslant \frac{1}{2} \epsilon_H \zeta \|\mathbf{d}_k\|^2 - \epsilon_H \|\mathbf{d}_k\|^2 + \delta_{g,k} \|\mathbf{d}_k\| && \text{(from (2.8c))} \\
&\leqslant -\frac{1}{2} \epsilon_H \|\mathbf{d}_k\|^2 + \frac{1-\zeta}{8} \max \left(\epsilon_g, \epsilon_H \|\mathbf{d}_k\| \right) \|\mathbf{d}_k\| && \text{(from } \zeta \in (0,1) \text{ and Condition 2.3)} \\
&= -\frac{1}{2} \epsilon_H \|\mathbf{d}_k\|^2 + \frac{1-\zeta}{8} \epsilon_H \|\mathbf{d}_k\|^2 && \text{(from } \|\mathbf{d}_k\| > \epsilon_g/\epsilon_H) \\
&< -\frac{3}{8} \epsilon_H \|\mathbf{d}_k\|^2.
\end{aligned}$$

We consider two cases here.

Case 1: Consider first the case in which the value $\alpha_k = 1$ is accepted by the backtracking line search procedure. We first note that in the case $\|\mathbf{g}_{k+1}\| - \delta_{g,k} - \delta_{g,k+1} \leqslant 0$, the claim (2.11) is satisfied trivially, because $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ and the right-hand side of (2.11) is 0. Thus we assume in the rest of

the argument for this case that $\|\mathbf{g}_{k+1}\| - \delta_{g,k} - \delta_{g,k+1} > 0$. We have

$$\begin{aligned}
\|\mathbf{g}_{k+1}\| &= \|\mathbf{g}_{k+1} - \mathbf{g}_k + \mathbf{g}_k\| \\
&= \|\mathbf{g}_{k+1} - \nabla f_{k+1} + \nabla f_{k+1} - \mathbf{g}_k - \nabla f_k + \nabla f_k - \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k - 2\epsilon_H \mathbf{d}_k + \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k - \mathbf{H}_k\mathbf{d}_k + \hat{\mathbf{r}}_k\| \\
&\leqslant \delta_{g,k} + \delta_{g,k+1} + \|\nabla f_{k+1} - \nabla f_k - \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k\| + \|2\epsilon_H \mathbf{d}_k\| + \|\nabla^2 f(\mathbf{x}_k)\mathbf{d}_k - \mathbf{H}_k\mathbf{d}_k\| + \|\hat{\mathbf{r}}_k\| \\
&\leqslant \delta_{g,k} + \delta_{g,k+1} + \frac{L_H}{2}\|\mathbf{d}_k\|^2 + 2\epsilon_H\|\mathbf{d}_k\| + \delta_H\|\mathbf{d}_k\| + \frac{1}{2}\epsilon_H\zeta\|\mathbf{d}_k\| \qquad \text{(from 2.8c)} \\
&= \delta_{g,k} + \delta_{g,k+1} + \left(2\epsilon_H + \delta_H + \frac{1}{2}\epsilon_H\zeta\right)\|\mathbf{d}_k\| + \frac{L_H}{2}\|\mathbf{d}_k\|^2 \\
&\leqslant \delta_{g,k} + \delta_{g,k+1} + \left(2\epsilon_H + \frac{1-\zeta}{2}\epsilon_H + \frac{1}{2}\epsilon_H\zeta\right)\|\mathbf{d}_k\| + \frac{L_H}{2}\|\mathbf{d}_k\|^2 \qquad \text{(from Condition 2.3)} \\
&= \delta_{g,k} + \delta_{g,k+1} + 2.5\epsilon_H\|\mathbf{d}_k\| + \frac{L_H}{2}\|\mathbf{d}_k\|^2.
\end{aligned}$$

We thus have $A\|\mathbf{d}_k\|^2 + B\|\mathbf{d}_k\| - C \geqslant 0$, where $A = L_H/2$, $B = 2.5\epsilon_H$ and $C = \|\mathbf{g}_{k+1}\| - \delta_{g,k} - \delta_{g,k+1} > 0$. Since for any $D \geqslant 0$ and $t \geqslant 0$ we have $-1 + \sqrt{1+Dt} \geqslant \left(-1 + \sqrt{1+D}\right)\min\{t, 1\}$ (see Royer & Wright, 2018, Lemma 17), it follows that

$$\begin{aligned}
\|\mathbf{d}_k\| \geqslant \frac{-B + \sqrt{B^2 + 4AC}}{2A} &= \left(\frac{-1 + \sqrt{1 + 4AC/B^2}}{2A}\right)B \geqslant \left(\frac{-1 + \sqrt{1+4A}}{2A}\right)\min\{C/B, B\} \\
&= \left(\frac{2}{\sqrt{1+4A}+1}\right)\min\{C/B, B\} \geqslant \left(\frac{1}{\sqrt{1+4A}}\right)\min\{C/B, B\},
\end{aligned}$$

where the last step follows from $A > 0$. By substituting for A, B and C, we obtain

$$\|\mathbf{d}_k\| \geqslant \frac{1}{\sqrt{1+2L_H}}\min\left\{\frac{\|\mathbf{g}_{k+1}\| - \delta_{g,k} - \delta_{g,k+1}}{2.5\epsilon_H}, 2.5\epsilon_H\right\}.$$

Since $\alpha_k = 1$ was accepted by the backtracking line search, we have

$$\begin{aligned}
f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{d}_k) &\geqslant \frac{\eta}{6}\|\mathbf{d}_k\|^3 \\
&\geqslant \frac{\eta}{6}\frac{1}{(1+2L_H)^{3/2}}\min\left\{\frac{(\|\mathbf{g}_{k+1}\| - \delta_{g,k} - \delta_{g,k+1})^3}{(2.5\epsilon_H)^3}, (2.5\epsilon_H)^3\right\}.
\end{aligned}$$

By combining this inequality with the trivial inequality obtained when $\|\mathbf{g}_{k+1}\| - \delta_{g,k} - \delta_{g,k+1} \leqslant 0$, we obtain (2.11) for the case of $\alpha_k = 1$.

Case 2: As a preliminary step, note that for any $\alpha \in [0, 1]$, we have the following:

$$\begin{aligned}
&\alpha \mathbf{g}_k^T \mathbf{d}_k + \tfrac{1}{2}\alpha^2 \mathbf{d}_k^T \mathbf{H}_k \mathbf{d}_k \\
&= \alpha \left[\hat{\mathbf{r}}_k - (\mathbf{H}_k + 2\epsilon_H I)\mathbf{d}_k\right]^T \mathbf{d}_k + \tfrac{1}{2}\alpha^2 \mathbf{d}_k^T \mathbf{H}_k \mathbf{d}_k \qquad \text{(from (2.9))} \\
&\leqslant \alpha \|\hat{\mathbf{r}}_k\| \|\mathbf{d}_k\| - \alpha \left(1 - \tfrac{1}{2}\alpha\right) \mathbf{d}_k^T (\mathbf{H}_k + 2\epsilon_H I)\mathbf{d}_k - \alpha^2 \epsilon_H \|\mathbf{d}_k\|^2 \\
&\leqslant \alpha \|\hat{\mathbf{r}}_k\| \|\mathbf{d}_k\| - \alpha \left(1 - \tfrac{1}{2}\alpha\right) \mathbf{d}_k^T (\mathbf{H}_k + 2\epsilon_H I)\mathbf{d}_k \\
&\leqslant \tfrac{1}{2}\alpha\epsilon_H \zeta \|\mathbf{d}_k\|^2 - \tfrac{1}{2}\alpha\epsilon_H \|\mathbf{d}_k\|^2 \qquad \text{(from } 1 - \tfrac{1}{2}\alpha \geqslant \tfrac{1}{2}\text{, (2.8a), and (2.8c))} \\
&= \tfrac{1}{2}\alpha\epsilon_H(\zeta - 1)\|\mathbf{d}_k\|^2. \qquad (2.12)
\end{aligned}$$

Now consider the case where $\alpha_k = 1$ is not accepted by the line search. In this case, suppose $j \geqslant 0$ is the largest integer such that the step acceptance condition is not satisfied. For this j, we have the following:

$$\begin{aligned}
&-\frac{\eta}{6}\theta^{3j}\|\mathbf{d}_k\|^3 \\
&\leqslant f(\mathbf{x}_k + \theta^j \mathbf{d}_k) - f(\mathbf{x}_k) \\
&\leqslant \theta^j \nabla f_k^T \mathbf{d}_k + \frac{\theta^{2j}}{2}\mathbf{d}_k^T \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k + \frac{L_H}{6}\theta^{3j}\|\mathbf{d}_k\|^3 \qquad \text{(from (2.2b))} \\
&\leqslant \theta^j \mathbf{g}_k^T \mathbf{d}_k + \frac{\theta^{2j}}{2}\mathbf{d}_k^T \mathbf{H}_k \mathbf{d}_k + \theta^j \delta_{g,k}\|\mathbf{d}_k\| + \frac{\theta^{2j}}{2}\delta_H \|\mathbf{d}_k\|^2 + \frac{L_H}{6}\theta^{3j}\|\mathbf{d}_k\|^3 \qquad \text{(from Definition 2.2)} \\
&\leqslant -\frac{\theta^j}{2}(1 - \zeta)\epsilon_H \|\mathbf{d}_k\|^2 + \theta^j \delta_{g,k}\|\mathbf{d}_k\| + \frac{\theta^{2j}}{2}\delta_H \|\mathbf{d}_k\|^2 + \frac{L_H}{6}\theta^{3j}\|\mathbf{d}_k\|^3 \qquad \text{(from 2.11)} \\
&\leqslant -\frac{\theta^j}{2}\|\mathbf{d}_k\|^2\big((1 - \zeta)\epsilon_H - \delta_H\big) + \theta^j \delta_{g,k}\|\mathbf{d}_k\| + \frac{L_H}{6}\theta^{3j}\|\mathbf{d}_k\|^3 \qquad \text{(from } 0 < \theta < 1\text{).}
\end{aligned}$$

By rearranging this expression, we obtain

$$\theta^{2j} \geqslant \left(\frac{3}{L_H + \eta}\right)\left(\frac{\big((1 - \zeta)\epsilon_H - \delta_H\big)\|\mathbf{d}_k\| - 2\delta_{g,k}}{\|\mathbf{d}_k\|^2}\right).$$

From Condition 2.2, we have $\delta_H \leqslant (1 - \zeta)\epsilon_H/2$, so this bound implies that

$$\theta^{2j} \geqslant \left(\frac{3}{L_H + \eta}\right)\frac{(1 - \zeta)\epsilon_H \|\mathbf{d}_k\| - 4\delta_{g,k}}{2\|\mathbf{d}_k\|^2}. \qquad (2.13)$$

Since by assumption $\|\mathbf{d}_k\| \geqslant \epsilon_g/\epsilon_H$, we have from Condition 2.2 that either

$$\delta_{g,k} \leqslant \frac{1-\zeta}{8}\epsilon_g = \frac{1-\zeta}{8}\epsilon_H \frac{\epsilon_g}{\epsilon_H} \leqslant \frac{(1-\zeta)\epsilon_H\|\mathbf{d}_k\|}{8}, \tag{2.14}$$

or else

$$\delta_{g,k} \leqslant \frac{1-\zeta}{8}\min(\epsilon_H\|\mathbf{d}_k\|, \|\mathbf{g}_k\|, \|\mathbf{g}_{k+1}\|) < \frac{(1-\zeta)\epsilon_H\|\mathbf{d}_k\|}{8}. \tag{2.15}$$

In either case, we have that $(1-\zeta)\epsilon_H\|\mathbf{d}_k\| - 4\delta_{g,k} \geqslant (1-\zeta)\epsilon_H\|\mathbf{d}_k\|/2$, so we have from (2.13) that

$$\theta^{2j} \geqslant \left(\frac{3}{L_H+\eta}\right)\left(\frac{(1-\zeta)\epsilon_H}{4\|\mathbf{d}_k\|}\right). \tag{2.16}$$

Since in the case under consideration, the acceptance condition for the backtracking line search fails for $j = 0$, the latter expression holds with $j = 0$, and we have

$$\|\mathbf{d}_k\| \geqslant \frac{3(1-\zeta)\epsilon_H}{4(L_H+\eta)}. \tag{2.17}$$

From (2.16), (2.8b) and (2.3), we know that

$$\theta^{2j} \geqslant \frac{3(1-\zeta)\epsilon_H}{4(L_H+\eta)}\|\mathbf{d}_k\|^{-1} \geqslant \frac{3(1-\zeta)\epsilon_H}{4(L_H+\eta)}\frac{\epsilon_H}{1.1U_g}. \tag{2.18}$$

Since

$$j_{\text{sol}} = \left\lceil \frac{1}{2}\log_\theta \frac{3(1-\zeta)\epsilon_H^2}{4.4U_g(L_H+\eta)} \right\rceil,$$

then for any $j > j_{\text{sol}}$, we have

$$\theta^{2j} < \theta^{2j_{\text{sol}}} \leqslant \frac{3(1-\zeta)\epsilon_H^2}{4.4U_g(L_H+\eta)}.$$

By comparing this expression with (2.18), we conclude that the line-search acceptance condition cannot be rejected for $j > j_{\text{sol}}$, so the step taken is $\alpha_k = \theta^{j_k}$ for some $j_k \leqslant j_{\text{sol}} + 1$. From (2.18), the preceding index $j = j_k - 1$ satisfies

$$\theta^{2j_k-2} \geqslant \frac{3(1-\zeta)\epsilon_H}{4(L_H+\eta)}\|\mathbf{d}_k\|^{-1},$$

so that

$$\theta^{j_k} \geqslant \sqrt{\frac{3\theta^2(1-\zeta)}{4(L_H+\eta)}}\epsilon_H^{1/2}\left\|\mathbf{d}_k\right\|^{-1/2}.$$

Then, we have

$$\begin{aligned} f(\mathbf{x}_k) - f(\mathbf{x}_k + \theta^{j_k}\mathbf{d}_k) &\geqslant \frac{\eta}{6}\theta^{3j_k}\left\|\mathbf{d}_k\right\|^3 \\ &\geqslant \frac{\eta}{6}\left[\frac{3\theta^2(1-\zeta)}{4(L_H+\eta)}\right]^{3/2}\epsilon_H^{3/2}\left\|\mathbf{d}_k\right\|^{3/2} \\ &\geqslant \frac{\eta}{6}\left[\frac{3\theta^2(1-\zeta)}{4(L_H+\eta)}\right]^{3/2}\epsilon_g^{3/2}, \end{aligned} \tag{2.19}$$

where the last inequality follows from $\|\mathbf{d}_k\| \geqslant \epsilon_g/\epsilon_H$.

We obtain the result by combining the two cases above. □

Next, we deal with the negative curvature directions, for which $d_{\text{type}} = \text{NC}$ and for which a backtracking birectional line search is used. Lemmas 2.6 and 2.7 bound the amount of decrease obtained from the negative curvature directions obtained in Procedures 1 and 2, respectively.

LEMMA 2.6 Suppose that Assumption 1 is satisfied and that Condition 2.2 holds for all k. Suppose that at iteration k of Algorithm 3, we have $\|\mathbf{g}_k\| \geqslant \epsilon_g$, so that Procedure 1 is called. When Procedure 1 outputs a direction $\mathbf{d}_k$ with $d_{\text{type}} = \text{NC}$ that is subsequently used as a search direction, the backtracking birectional line search terminates with (2.8) satisfied by either $\alpha_k = \theta^{j_k}$ or $\alpha_k = -\theta^{j_k}$, with $j_k \leqslant j_{\text{nc}} + 1$, where

$$j_{\text{nc}} = \left\lceil \log_\theta \frac{3}{2(L_H+\eta)} \right\rceil.$$

The resulting step $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{d}_k$ satisfies

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant c_{\text{nc}}\epsilon_H^3,$$

where

$$c_{\text{nc}} = \frac{\eta}{6}\min\left\{\left[\frac{3\theta}{2(L_H+\eta)}\right]^3, 1\right\}.$$

Proof. Note first that by (2.10), we have $\|\mathbf{d}_k\| = |\mathbf{d}^T\mathbf{H}_k\mathbf{d}| \geqslant \epsilon_H$. Thus, if $\alpha_k = \pm 1$, we have by (2.8) that $f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant \frac{\eta}{6}\|\mathbf{d}_k\|^3 \geqslant \frac{\eta}{6}\epsilon_H^3$, so the result holds in this case.

When $|\alpha_k| < 1$, using (2.10) again, we have

$$\mathbf{d}_k^T\mathbf{H}_k\mathbf{d}_k = -\|\mathbf{d}_k\|^3 \leqslant -\epsilon_H\|\mathbf{d}_k\|^2.$$

We have from Definition 2.1 that

$$|\mathbf{d}_k^T(\mathbf{H}_k - \nabla^2 f(\mathbf{x}_k))\mathbf{d}_k| \leqslant \delta_H \|\mathbf{d}_k\|^2,$$

so by combining the last two expressions, we have

$$\mathbf{d}_k^T \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k \leqslant -\|\mathbf{d}_k\|^3 + \delta_H \|\mathbf{d}_k\|^2. \tag{2.20}$$

Let $j \geqslant 0$ be an integer such that neither θ^j nor $-\theta^j$ satisfies the criterion (2.8). Supposing first that $\nabla f(\mathbf{x}_k)^T \mathbf{d}_k \leqslant 0$, we have from (2.2b) and (2.20) that

$$\begin{aligned} -\frac{\eta}{6}\theta^{3j}\|\mathbf{d}_k\|^3 &\leqslant f(\mathbf{x}_k + \theta^j \mathbf{d}_k) - \mathbf{f}(\mathbf{x}_k) \\ &\leqslant \theta^j \nabla f(\mathbf{x}_k)^T \mathbf{d}_k + \frac{\theta^{2j}}{2}\mathbf{d}_k^T \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k + \frac{L_H}{6}\theta^{3j}\|\mathbf{d}_k\|^3 \\ &\leqslant -\frac{\theta^{2j}}{2}\|\mathbf{d}_k\|^3 + \frac{\theta^{2j}}{2}\delta_H\|\mathbf{d}_k\|^2 + \frac{L_H}{6}\theta^{3j}\|\mathbf{d}_k\|^3. \end{aligned} \tag{2.21}$$

Supposing instead that $\nabla f(\mathbf{x}_k)^T \mathbf{d}_k > 0$, we have by considering the step $-\theta^j$ that

$$\begin{aligned} -\frac{\eta}{6}\theta^{3j}\|\mathbf{d}_k\|^3 &\leqslant f(\mathbf{x}_k - \theta^j \mathbf{d}_k) - \mathbf{f}(\mathbf{x}_k) \\ &\leqslant -\theta^j \nabla f(\mathbf{x}_k)^T \mathbf{d}_k + \frac{\theta^{2j}}{2}\mathbf{d}_k^T \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k + \frac{L_H}{6}\theta^{3j}\|\mathbf{d}_k\|^3 \\ &\leqslant -\frac{\theta^{2j}}{2}\|\mathbf{d}_k\|^3 + \frac{\theta^{2j}}{2}\delta_H\|\mathbf{d}_k\|^2 + \frac{L_H}{6}\theta^{3j}\|\mathbf{d}_k\|^3, \end{aligned}$$

yielding the same inequality as (2.21). After rearrangement of this inequality and using $\|\mathbf{d}_k\| \geqslant \epsilon_H$, it follows that

$$\theta^j \geqslant \left(\frac{6}{L_H + \eta}\right)\left(\frac{\|\mathbf{d}_k\| - \delta_H}{2\|\mathbf{d}_k\|}\right) = \frac{3}{L_H + \eta} - \frac{3\delta_H}{(L_H + \eta)\|\mathbf{d}_k\|} \geqslant \frac{3}{L_H + \eta} - \frac{3\delta_H}{(L_H + \eta)\epsilon_H}. \tag{2.22}$$

Since from Condition 2.2, we have $\delta_H \leqslant (1 - \zeta)\epsilon_H/4 < \epsilon_H/4$, then

$$\theta^j \geqslant \frac{3}{2(L_H + \eta)}. \tag{2.23}$$

Meanwhile, we have for $j > j_{\text{nc}}$ that

$$\theta^j < \theta^{j_{\text{nc}}} \leqslant \frac{3}{2(L_H + \eta)}.$$

The last two inequalities together imply that $j \leqslant j_{\text{nc}}$, so the line search must terminate with $\alpha_k = \pm\theta^{j_k}$ for some $j_k \leqslant j_{\text{nc}} + 1$. Since (2.23) must hold for $j = j_k - 1$, we have

$$\theta^{j_k-1} \geqslant \frac{3}{2(L_H+\eta)} \implies |\alpha_k| = \theta^{j_k} \geqslant \frac{3\theta}{2(L_H+\eta)}.$$

Thus, from the step acceptance condition (2.8) together with (2.10) and the definition of c_{nc}, we have

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant \frac{\eta}{6}|\alpha_k|^3\|\mathbf{d}_k\|^3 \geqslant c_{\text{nc}}\epsilon_H^3,$$

so the required claim also holds in the case of $|\alpha_k| < 1$, completing the proof. □

We now turn our attention to the property of Procedure 2. The following lemma shows that when a negative curvature direction is obtained from Procedure 2, we can guarantee descent in the function in a similar fashion to Lemma 2.6.

LEMMA 2.7 Suppose that Assumption 1 is satisfied and that Condition 2.2 holds for all k. Suppose that at iteration k of Algorithm 3, the search direction $\mathbf{d}_k$ is a negative curvature direction for $\mathbf{H}_k$, obtained from Procedure 2. Then the backtracking bidirectional line search terminates with step size either $\alpha_k = \theta^{j_k}$ or $\alpha_k = -\theta^{j_k}$ with $j_k \leqslant j_{\text{nc}} + 1$, where j_{nc} is defined as in Lemma 2.6. Moreover, the decrease in function value resulting from the chosen step size satisfies

$$f(\mathbf{x}_k) - f(\mathbf{x}_k + \alpha_k\mathbf{d}_k) \geqslant \frac{c_{\text{nc}}}{8}\epsilon_H^3, \tag{2.24}$$

where c_{nc} is defined in Lemma 2.6.

Proof. Note that

$$\mathbf{d}_k^T\mathbf{H}\mathbf{d}_k \leqslant -\|\mathbf{d}_k\|^3 \leqslant -\frac{\epsilon_H}{2}\|\mathbf{d}_k\|^2,$$

so that $\|\mathbf{d}_k\| \geqslant \epsilon_H/2$. In the first part of the proof, for the case $\alpha_k = \pm 1$, we have

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant \frac{\eta}{6}\|\mathbf{d}_k\|^3 \geqslant \frac{\eta}{6}\frac{1}{8}\epsilon_H^3 \geqslant \frac{c_{\text{nc}}}{8}\epsilon_H^3,$$

so the result holds in this case. The analysis of the case $|\alpha_k| < 1$ proceeds as in the proof of Lemma 2.6 until the lower bound on θ^j in (2.22), where because of $\|\mathbf{d}_k\| \geqslant \epsilon_H/2$, we have

$$\theta^j \geqslant \frac{3}{L_H+\eta} - \frac{6\delta_H}{(L_H+\eta)\epsilon_H},$$

which, because of $\delta_H \leqslant \epsilon_H/4$, still yields the lower bound (2.23), allowing the result of the proof to proceed as in the earlier result, except for the factor of 1/8. □

Now comes a crucial step. When the output direction $\mathbf{d}_k$ from Procedure 1 satisfies $\|\mathbf{d}_k\| \leqslant \epsilon_g/\epsilon_H$ and Procedure 2 detects no significant negative curvature in the Hessian, the update of $\mathbf{x}_k$ with unit step

along $\mathbf{d}_k$ is the final step of Algorithm 3. Dealing with this case is critical to obtaining the convergence rate of our inexact damped Newton-CG algorithm.

LEMMA 2.8 Suppose that Assumption 1 is satisfied and that Condition 2.2 holds for all k. Suppose that Algorithm 3 terminates at iteration k at line 12, and returns $\mathbf{x}_k + \mathbf{d}_k$, where $\mathbf{d}_k$ is obtained from Procedure 1 and satisfies $\|\mathbf{d}_k\| \leqslant \epsilon_g/\epsilon_H$. Then we have

$$\|\nabla f(\mathbf{x}_k + \mathbf{d}_k)\| \leqslant \frac{L_H}{2}\frac{\epsilon_g^2}{\epsilon_H^2} + 4\epsilon_g.$$

If in addition the property $\mathbf{H}_k \succeq -\epsilon_H I$ holds, then

$$\lambda_{\min}(\nabla^2 f(\mathbf{x}_k + \mathbf{d}_k)) \geqslant -\left(\frac{5}{4}\epsilon_H + L_H\frac{\epsilon_g}{\epsilon_H}\right) I.$$

Proof. Note that termination at line 12 occurs only if $d_{\text{type}} = \text{SOL}$, so Part 1 of Lemma 2.4 holds. For the gradient norm at $\mathbf{x}_k + \mathbf{d}_k$, we have

$$\begin{aligned}
\|\nabla f(\mathbf{x}_k + \mathbf{d}_k)\| &\leqslant \left\|\nabla f(\mathbf{x}_k + \mathbf{d}_k) - \nabla f(\mathbf{x}_k) - \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k + \mathbf{H}_k\mathbf{d}_k + \mathbf{g}_k\right\| \\
&\quad + \|\nabla f(\mathbf{x}_k) - \mathbf{g}_k\| + \left\|\nabla^2 f(\mathbf{x}_k)\mathbf{d}_k - \mathbf{H}_k\mathbf{d}_k\right\| \\
&\leqslant \left\|\nabla f(\mathbf{x}_k + \mathbf{d}_k) - \nabla f(\mathbf{x}_k) - \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k\right\| + \|\mathbf{H}_k\mathbf{d}_k + \mathbf{g}_k\| + \delta_{g,k} + \delta_H\|\mathbf{d}_k\| \\
&\leqslant \left\|\nabla f(\mathbf{x}_k + \mathbf{d}_k) - \nabla f(\mathbf{x}_k) - \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k\right\| + \|\hat{\mathbf{r}}_k\| + 2\epsilon_H\|\mathbf{d}_k\| + \delta_{g,k} + \delta_H\|\mathbf{d}_k\| \\
&\leqslant \frac{L_H}{2}\|\mathbf{d}_k\|^2 + \frac{1}{2}\epsilon_H\zeta\|\mathbf{d}_k\| + (2\epsilon_H + \delta_H)\|\mathbf{d}_k\| + \delta_{g,k} \qquad \text{(from 2.2a and 2.8c)} \\
&\leqslant \frac{L_H}{2}\|\mathbf{d}_k\|^2 + 3\epsilon_H\|\mathbf{d}_k\| + \delta_{g,k} \qquad \text{(since } \zeta \in (0,1) \text{ and } \delta_H \leqslant \epsilon_{H/2}) \\
&\leqslant \frac{L_H}{2}\|\mathbf{d}_k\|^2 + 3\epsilon_H\|\mathbf{d}_k\| + \left(\frac{1-\zeta}{8}\right)\max\left(\epsilon_g, \min(\epsilon_H\|\mathbf{d}_k\|, \|\mathbf{g}_k\|, \|\mathbf{g}_{k+1}\|)\right) \\
&\leqslant \frac{L_H}{2}\frac{\epsilon_g^2}{\epsilon_H^2} + 3\epsilon_H\|\mathbf{d}_k\| + \frac{1-\zeta}{8}\max\left(\epsilon_g, \epsilon_H\|\mathbf{d}_k\|\right) \\
&\leqslant \frac{L_H}{2}\frac{\epsilon_g^2}{\epsilon_H^2} + 3\epsilon_g + \frac{1-\zeta}{8}\epsilon_g \qquad \text{(since } \|\mathbf{d}_k\| \leqslant \epsilon_g/\epsilon_H) \\
&\leqslant \frac{L_H}{2}\frac{\epsilon_g^2}{\epsilon_H^2} + 4\epsilon_g,
\end{aligned}$$

as required.

For the second-order condition, since $\mathbf{H}_k \succeq -\epsilon_H I$ and $\delta_H \leqslant \epsilon_{H/4}$ (from Condition 2.2), we have

$$\nabla^2 f(\mathbf{x}_k + \mathbf{d}_k) \succeq \nabla^2 f(\mathbf{x}_k) - L_H \|\mathbf{d}_k\| I \succeq \mathbf{H}_k - \delta_H I - L_H \frac{\epsilon_g}{\epsilon_H} I \succeq -\left(\frac{5}{4}\epsilon_H + L_H \frac{\epsilon_g}{\epsilon_H}\right) I.$$

This completes the proof. □

Now, combining Lemmas 2.5–2.8, we obtain the iteration complexity for Algorithm 3.

THEOREM 2.3 Suppose that Assumption 1 is satisfied and that Condition 2.2 holds for all k. For a given $\epsilon > 0$, let $\epsilon_H = \sqrt{L_H \epsilon}, \epsilon_g = \epsilon$. Define

$$\bar{K} := \left\lceil \frac{3(f(\mathbf{x}_0) - f_{\text{low}})}{\min\left(\frac{1}{64 L_H^{3/2}} c_{\text{sol}}, 8 L_H^{3/2} c_{\text{sol}}, L_H^{3/2} c_{\text{nc}}/8\right)} \epsilon^{-3/2} \right\rceil + 5, \tag{2.25}$$

where c_{sol} and c_{nc} are defined in Lemmas 2.5 and 2.6, respectively. Then Algorithm 3 terminates in at most $\bar{K}$ iterations at a point, satisfying

$$\|\nabla f(\mathbf{x})\| \lesssim \epsilon.$$

Moreover, with probability at least $(1-\delta)^{\bar{K}}$ the point returned by Algorithm 3 also satisfies the approximate second-order condition

$$\lambda_{\min}(\nabla^2 f(\mathbf{x})) \gtrsim -\sqrt{L_H \epsilon}. \tag{2.26}$$

Here, $\lesssim$ and $\gtrsim$ denote that the corresponding inequality holds up to a certain constant that is independent of ϵ and L_H.

Proof. Note first that for our choices of ϵ_g and ϵ_H, the threshold ϵ_g/ϵ_H for $\|\mathbf{d}_k\|$ in line 9 of Algorithm 3 becomes $\sqrt{\epsilon/L_H}$.

We show first that Algorithm 3 terminates after at most $\bar{K}$ steps. We taxonomize the iterations into five classes. To specify these classes, we denote by $\mathbf{d}_k$ and d_{type} the values of these variables *immediately before a step is taken or termination is declared*, bearing in mind that these variables can be reassigned during iteration k, in Line 14. Supposing for contradiction that Algorithm 3 runs for at least K steps, for some $K > \bar{K}$, we define the five classes of indices as follows.

$$\begin{aligned}
\mathcal{K}_1 &:= \{k = 0, 1, 2, \ldots, K-1 \mid \|\mathbf{g}_k\| < \epsilon\} \\
\mathcal{K}_2 &:= \{k = 0, 1, 2, \ldots, K-1 \mid \|\mathbf{g}_k\| \geqslant \epsilon,\ d_{\text{type}} = \text{SOL},\ \|\mathbf{d}_k\| > \sqrt{\epsilon/L_H},\ \|\mathbf{g}_{k+1}\| < \epsilon\} \\
\mathcal{K}_3 &:= \{k = 0, 1, 2, \ldots, K-1 \mid \|\mathbf{g}_k\| \geqslant \epsilon,\ d_{\text{type}} = \text{SOL},\ \|\mathbf{d}_k\| > \sqrt{\epsilon/L_H},\ \|\mathbf{g}_{k+1}\| \geqslant \epsilon\} \\
\mathcal{K}_4 &:= \{k = 0, 1, 2, \ldots, K=1 \mid \|\mathbf{g}_k\| \geqslant \epsilon,\ d_{\text{type}} = \text{SOL},\ \|\mathbf{d}_k\| \leqslant \sqrt{\epsilon/L_H}\} \\
\mathcal{K}_5 &:= \{k = 0, 1, 2, \ldots, K-1 \mid \|\mathbf{g}_k\| \geqslant \epsilon,\ d_{\text{type}} = \text{NC}\}.
\end{aligned}$$

Obviously, $K = |\mathscr{K}_1| + |\mathscr{K}_2| + |\mathscr{K}_3| + |\mathscr{K}_4| + |\mathscr{K}_5|$. We consider each of these types of steps in turn.

Case 1: $k \in \mathscr{K}_1$. The update $\mathbf{d}_k$ in this case must come from Procedure 2. Either the method terminates (which happens at most once!) or from Lemma 2.7, we have that

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant \frac{1}{8} c_{\mathrm{nc}} \epsilon_H^3 = \frac{1}{8} L_H^{3/2} c_{\mathrm{nc}} \epsilon^{3/2}. \tag{2.27}$$

Thus the total amount of decrease that results from steps in $\mathscr{K}_1$ is at least $\big(|\mathscr{K}_1| - 1\big) L_H^{3/2} c_{\mathrm{nc}} \epsilon^{3/2}/8$.

Case 2: $k \in \mathscr{K}_2$. With Lemma 2.5, we can guarantee only that $f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant 0$. However, since $\|\mathbf{g}_{k+1}\| < \epsilon$, the *next* iterate must belong to class $\mathscr{K}_1$. Therefore we have $|\mathscr{K}_2| \leqslant |\mathscr{K}_1|$.

Case 3: $k \in \mathscr{K}_3$. Here the step $\mathbf{d}_k$ is an approximate solution of the damped Newton equations, and we can apply Lemma 2.5 to obtain a nontrivial lower bound on the decrease in f. By Condition 2.2, we have

$$\begin{aligned}
\delta_{g,k} &\leqslant \frac{1}{8} \max\Big(\epsilon_g, \min(\epsilon_H \|\mathbf{d}_k\|, \|\mathbf{g}_k\|, \|\mathbf{g}_{k+1}\|)\Big) \leqslant \frac{1}{8} \max(\epsilon_g, \|\mathbf{g}_{k+1}\|) = \frac{1}{8} \|\mathbf{g}_{k+1}\|, \\
\delta_{g,k+1} &\leqslant \frac{1}{8} \max\Big(\epsilon_g, \min(\epsilon_H \|\mathbf{d}_{k+1}\|, \|\mathbf{g}_{k+1}\|, \|\mathbf{g}_{k+2}\|)\Big) \leqslant \frac{1}{8} \max(\epsilon_g, \|\mathbf{g}_{k+1}\|) = \frac{1}{8} \|\mathbf{g}_{k+1}\|,
\end{aligned}$$

so that

$$\|\mathbf{g}_{k+1}\| - \delta_{g,k} - \delta_{g,k+1} \geqslant \frac{3}{4} \|\mathbf{g}_{k+1}\| \geqslant \frac{3}{4} \epsilon_g = \frac{3}{4} \epsilon.$$

Thus, from (2.11) in Lemma 2.5, we have for this type of step that

$$\begin{aligned}
f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) &\geqslant c_{\mathrm{sol}} \max\left\{0, \min\left(\frac{(\|\mathbf{g}_{k+1}\| - \delta_{g,k} - \delta_{g,k+1})^3}{(2.5\epsilon_H)^3}, (2.5\epsilon_H)^3, \epsilon_g^{3/2}\right)\right\} \\
&\geqslant c_{\mathrm{sol}} \min\left(\frac{(\frac{3}{4}\epsilon)^3}{(2.5\sqrt{L_H \epsilon})^3}, (2.5\sqrt{L_H \epsilon})^3, \epsilon^{3/2}\right) \\
&= c_{\mathrm{sol}} \min\left(\frac{1}{64 L_H^{3/2}}, 8 L_H^{3/2}, 1\right) \epsilon^{3/2} = c_{\mathrm{sol}} \min\left(\frac{1}{64 L_H^{3/2}}, 8 L_H^{3/2}\right) \epsilon^{3/2}.
\end{aligned}$$

Case 4: $k \in \mathscr{K}_4$. In this case, Procedure 1 outputs $d_{\mathrm{type}} = \mathrm{SOL}$ along with a 'small' value of $\mathbf{d}_k$. Subsequently, Procedure 2 was called, but it must have returned with a certification of near-positive-definiteness of $\mathbf{H}_k$, since d_{type} was not switched to NC. Thus, according to Lemma 2.8, termination occurs with output $\mathbf{x}_k + \mathbf{d}_k$. Thus, this case can occur at most once, and we have $|\mathscr{K}_4| \leqslant 1$.

Case 5: $k \in \mathscr{K}_5$. In this case, either the algorithm terminates and outputs $\mathbf{x} = \mathbf{x}_k$ (which happens at most once), or else a step is taken along a negative curvature direction for $\mathbf{H}_k$, detected either in Procedure 1 or Procedure 2. In the former case (detection in Procedure 1), we have from Lemma 2.6

that $f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant c_{\mathrm{nc}}\epsilon_H^3 = c_{\mathrm{nc}}L_H^{3/2}\epsilon^{3/2}$, while in the latter case (detection in Procedure 2), we have from Lemma 2.7 that $f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant \frac{1}{8}L_H^{3/2}c_{\mathrm{nc}}\epsilon^{3/2}$. Thus, the total decrease in f resulting from steps of this class is bounded below by $(|\mathcal{K}_5| - 1)\frac{1}{8}L_H^{3/2}c_{\mathrm{nc}}\epsilon^{3/2}$.

The total decrease of f over all K steps cannot exceed $f(\mathbf{x}_0) - f_{\mathrm{low}}$. We thus have

$$\begin{aligned} f(\mathbf{x}_0) - f_{\mathrm{low}} &\geqslant \sum_{k=0}^{K-1}(f(\mathbf{x}_k) - f(\mathbf{x}_{k+1})) \\ &\geqslant \sum_{k\in\mathcal{K}_1}(f(\mathbf{x}_k) - f(\mathbf{x}_{k+1})) + \sum_{k\in\mathcal{K}_3}(f(\mathbf{x}_k) - f(\mathbf{x}_{k+1})) + \sum_{k\in\mathcal{K}_5}(f(\mathbf{x}_k) - f(\mathbf{x}_{k+1})) \\ &\geqslant \big(|\mathcal{K}_1| + |\mathcal{K}_5| - 2\big)\frac{1}{8}L_H^{3/2}c_{\mathrm{nc}}\epsilon^{3/2} + |\mathcal{K}_3|\, c_{\mathrm{sol}} \min\left(\frac{1}{64L_H^{3/2}}, 8L_H^{3/2}\right)\epsilon^{3/2}. \end{aligned}$$

Therefore, we have

$$\begin{aligned} |\mathcal{K}_1| + |\mathcal{K}_5| - 2 &\leqslant \frac{f(\mathbf{x}_0) - f_{\mathrm{low}}}{L_H^{3/2}c_{\mathrm{nc}}/8}\epsilon^{-3/2}, \\ |\mathcal{K}_3| &\leqslant \frac{f(\mathbf{x}_0) - f_{\mathrm{low}}}{c_{\mathrm{sol}} \min\left(\frac{1}{64L_H^{3/2}}, 8L_H^{3/2}\right)}\epsilon^{-3/2}. \end{aligned}$$

Finally, we have

$$\begin{aligned} K &= |\mathcal{K}_1| + |\mathcal{K}_2| + |\mathcal{K}_3| + |\mathcal{K}_4| + |\mathcal{K}_5| \\ &\leqslant 2|\mathcal{K}_1| + |\mathcal{K}_3| + 1 + |\mathcal{K}_5| \\ &\leqslant 2\big(|\mathcal{K}_1| + |\mathcal{K}_5| - 2\big) + |\mathcal{K}_3| + 5 \\ &\leqslant \frac{2(f(\mathbf{x}_0) - f_{\mathrm{low}})}{L_H^{3/2}c_{\mathrm{nc}}/8}\epsilon^{-3/2} + \frac{f(\mathbf{x}_0) - f_{\mathrm{low}}}{c_{\mathrm{sol}} \min\left(\frac{1}{64L_H^{3/2}}, 8L_H^{3/2}\right)}\epsilon^{-3/2} + 5 \\ &\leqslant \frac{3(f(\mathbf{x}_0) - f_{\mathrm{low}})}{\min\left(\frac{1}{64L_H^{3/2}}c_{\mathrm{sol}}, 8L_H^{3/2}c_{\mathrm{sol}}, L_H^{3/2}c_{\mathrm{nc}}/8\right)}\epsilon^{-3/2} + 5 \leqslant \bar{K}, \end{aligned}$$

which contradicts our assertion that $K > \bar{K}$. Thus Algorithm 3 terminates in at most $\bar{K}$ steps.

Note that if termination occurs at Line 24 of Algorithm 3, the returned value of $\mathbf{x} = \mathbf{x}_k$ certainly has $\|\nabla f(\mathbf{x})\| \lesssim \epsilon_g = \epsilon$. This is because when $\|\mathbf{g}_k\| \leqslant \epsilon_g$, we have from Condition 2.2 that $\delta_{g,k} \leqslant (1 - \zeta)\epsilon_g/8$, so that $\|\nabla f(\mathbf{x})\| \leqslant \|\mathbf{g}_k\| + \delta_{g,k} \lesssim \epsilon_g$. Alternatively, if termination occurs at Line 12, for

the returned value of $\mathbf{x} = \mathbf{x}_k + \mathbf{d}_k$, we have

$$\|\nabla f(\mathbf{x})\| \leqslant \frac{L_H}{2}\frac{\epsilon_g^2}{\epsilon_H^2} + 4\epsilon_g = \frac{L_H}{2}\frac{\epsilon^2}{L_H\epsilon} + 4\epsilon = \frac{9}{2}\epsilon.$$

Thus, the claim $\|\nabla f(\mathbf{x})\| \lesssim \epsilon$ at the termination point $\mathbf{x}$ holds.

We now verify the claims about probability of failure and the second-order conditions. Note that for both types of termination (at Lines 12 and 24 of Algorithm 3), Procedure 2 issues a certificate that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon_H$. Subject to this certificate being correct, we show now that our claim (2.26) holds. When termination occurs at line 12, we have in this case from Lemma 2.8 that at the returned point $\mathbf{x} = \mathbf{x}_k + \mathbf{d}_k$, we have

$$\lambda_{\min}(\nabla^2 f(\mathbf{x})) \geqslant -\left(\frac{5}{4}\epsilon_H + L_H\frac{\epsilon_g}{\epsilon_H}\right) = -\frac{9}{4}\sqrt{L_H\epsilon},$$

as required. For termination at Line 24, we have directly that $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \geqslant -\epsilon_H = -\sqrt{L_H\epsilon}$, again verifying the claim.

We now calculate a bound on the probability of incorrect termination, which can occur at either Line 12 or Line 24 when Procedure 2 issues a certificate that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon_H$, whereas in fact $\lambda_{\min}(\mathbf{H}_k) < -\epsilon_H$. The proof is a simple adaptation from Xie & Wright (2021, Theorem 2) and Curtis *et al.* (2021, Theorem 4.6), the adaptations for inexactness being fairly straightforward. We include the argument here for the sake of completeness. The possibility of such an event happening on any individual call to Procedure 2 is bounded above by δ. For all iterates k, we denote by $\tilde{P}_k$ the probability that Algorithm 3 reaches iteration k, but $\lambda_{\min}(\mathbf{H}_k) < -\epsilon_H$, and denote by P_k the probability that Algorithm 3 reaches iteration k, but $\lambda_{\min}(\mathbf{H}_k) < -\epsilon_H$, yet the algorithm terminates due to Procedure 2 issuing an incorrect certificate. Clearly, we have $P_k \leqslant \delta\tilde{P}_k$ for all $k = 0, 1, \ldots, \bar{K}$. Since it is trivially true for all k that

$$\tilde{P}_k + \sum_{i=0}^{k-1} P_i \leqslant 1,$$

we have for all k that

$$P_k \leqslant \delta\tilde{P}_k \leqslant \delta\left(1 - \sum_{i=0}^{k-1} P_i\right). \tag{2.28}$$

Now let M_k be the total number of calls to Procedure 2 that have occurred up to and including iteration k of Algorithm 3. We prove by induction that $\sum_{i=0}^{k} P_i \leqslant 1 - (1-\delta)^{M_k}$ for all k. For $k = 0$, the claim holds trivially, both in the case of $M_0 = 0$ (in which case $P_0 = 0$) and $M_0 = 1$ (in which case $P_0 \leqslant \delta$). Supposing now that the claim is true for some $k \geqslant 0$, we show that it continues to hold for $k+1$. If Algorithm 3 reaches iteration $k+1$ with $\lambda_{\min}(\mathbf{H}_{k+1}) < -\epsilon_H$, and Procedure 2 is *not* called at this

iteration, then $M_{k+1} = M_k$ and $P_{k+1} = 0$, so by the induction hypothesis, we have

$$\sum_{i=0}^{k+1} P_i = \sum_{i=0}^{k} P_i \leqslant 1 - (1-\delta)^{M_k} = 1 - (1-\delta)^{M_{k+1}},$$

as required. In the other case in which Algorithm 3 reaches iteration $k+1$ with $\lambda_{\min}(\mathbf{H}_{k+1}) < -\epsilon_H$, and Procedure 2 *is* called at this iteration, then $M_{k+1} = M_k + 1$, so by using (2.28) and the inductive hypothesis, we have

$$\begin{aligned}
\sum_{i=0}^{k+1} P_i &= \sum_{i=0}^{k} P_i + P_{k+1} \\
&\leqslant \sum_{i=0}^{k} P_i + \delta\left(1 - \sum_{i=0}^{k} P_i\right) \\
&= \delta + (1-\delta)\sum_{i=0}^{k} P_i \\
&\leqslant \delta + (1-\delta)\left(1 - (1-\delta)^{M_k}\right) \\
&= 1 - (1-\delta)^{M_k+1} = 1 - (1-\delta)^{M_{k+1}},
\end{aligned}$$

as required. Since $M_k \leqslant k \leqslant \bar{K}$ for all $k = 1, 2, \ldots, \bar{K}$, we have that the probability that Algorithm 3 terminates incorrectly on *any* iteration is bounded above by $1 - (1-\delta)^{\bar{K}}$. So when termination occurs, the condition (2.26) holds at the termination point with probability at least $(1-\delta)^{\bar{K}}$, as claimed. □

By incorporating the complexity of Procedures 1 and 2, as described in Lemmas 2.1 and 2.3, we can obtain an upper bound on the number of approximate gradient and approximate Hessian-vector product evaluations required during a run of Algorithm 3. The iteration count for the algorithm is bounded by $\mathcal{O}(\epsilon^{-3/2})$ in Theorem 2.3 and each iteration requires one approximate gradient evaluation. Additionally, each iteration of Algorithm 3 may require a call to Procedure 1, which by Lemma 2.1 requires $\tilde{\mathcal{O}}(\epsilon_H^{-1/2}) = \tilde{\mathcal{O}}(\epsilon^{-1/4})$ approximate Hessian-vector products. A call to Procedure 2 may also be required on some iterations. Here, by Lemma 2.3, $\mathcal{O}(\epsilon_H^{-1/2}) = \mathcal{O}(\epsilon^{-1/4})$ approximate Hessian-vector products may be required also. We summarize these observations in the following corollary.

COROLLARY 2.1 Suppose that the assumptions of Theorem 2.3 hold. Let $\epsilon_g = \epsilon, \epsilon_H = \sqrt{L_H \epsilon}$, and $\bar{K}$ be defined as in (2.25). Then for d sufficiently large relative to $\epsilon^{-1/2}$, Algorithm 3 terminates after at most $\tilde{\mathcal{O}}(\epsilon^{-7/4})$ matrix-vector products with the approximate Hessians and at most $\mathcal{O}(\epsilon^{-3/2})$ evaluations of approximate gradients. With probability at least $(1-\delta)^{\bar{K}}$, it returns a point that satisfies the approximate first- and second-order conditions described in Theorem 2.3.

2.4 *Inexact Newton-CG algorithm without line search*

Although Algorithm 3 employs approximate gradients and Hessian at various steps, the use of backtracking line search to compute the step-size α_k requires exact evaluations of the function f and its gradient. This setting has indeed been considered in some previous work, e.g., Roosta & Mahoney

(2019); Yao *et al.* (2020). When gradient evaluation has similar computational cost to the corresponding function evaluation, we may not save much in computation by requiring only an approximate gradient. We show in this section that a pre-defined ('fixed') value of the step length α_k can be carefully chosen to obviate the need for function evaluations. The advantage of not requiring exact evaluations of functions is considerable, but there are disadvantages too. First, the computed fixed step size is conservative, so the guaranteed descent in the objective generally will be smaller than in Algorithm 3; see Lemmas 2.5 to 2.7. Secondly, our approach makes use of an approximate upper bound L_H on the Lipschitz constant of the Hessian, which might not be readily available. Fortunately, there are many important instances (especially in machine learning) where an estimate of L_H can be obtained easily; for example, empirical risk minimization problems involving the squared loss (Xu *et al.*, 2020b) and Welsch's exponential variant (Zhang *et al.*, 2019). See Table 1 for details.

We state our variant of the Inexact Newton-CG Algorithm that does not require line search as Algorithm 4. Lines 6, 14, 24 and 27–31 constitute the main differences between Algorithms 3 and 4.

The analysis of this section makes use of the following condition.

CONDITION 2.4 The inexact gradient $\mathbf{g}_k$ and Hessian $\mathbf{H}_k$ satisfy Condition 2.1 with

$$\delta_{g,k} \leqslant \frac{1-\zeta}{8} \min\left(\frac{3\epsilon_H^2}{65(L_H+\eta)}, \max\left(\epsilon_g, \min(\epsilon_H \|\mathbf{d}_k\|, \|\mathbf{g}_k\|, \|\mathbf{g}_{k+1}\|)\right)\right)$$

$$\text{and} \quad \delta_H \leqslant \frac{1-\zeta}{4}\epsilon_H.$$

Throughout this section, we fix $\epsilon_H = \sqrt{L_H \epsilon_g}$, so that $\epsilon_g/\epsilon_H = \sqrt{\epsilon_g/L_H}$.

In the next three lemmas, we show that the choices of α_k in Algorithm 4 lead to the step length acceptance condition used in Algorithm 3 being satisfied, that is,

$$-\frac{\eta}{6}\alpha_k^3 \|\mathbf{d}_k\|^3 \geqslant f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) - f(\mathbf{x}_k). \tag{2.29}$$

We now show that the fixed step size can result in a sufficient descent in the function $f(\mathbf{x}_k)$ when $d_{\text{type}} = \text{SOL}$ and $\|\mathbf{d}_k\| \geqslant \sqrt{\epsilon_g/L_H}$. The following lemma can be viewed as a modification of Lemma 2.5 with fixed step size.

LEMMA 2.9 Suppose that Assumption 1 is satisfied and that Condition 2.4 holds for all k. Suppose that at iteration k of Algorithm 4, we have $\|\mathbf{g}_k\| \geqslant \epsilon_g$, so that Procedure 1 is called. When Procedure 1 outputs a direction $\mathbf{d}_k$ with $d_{\text{type}} = \text{SOL}$ and $\|\mathbf{d}_k\| \geqslant \epsilon_g/\epsilon_H$, Algorithm 4 sets

$$\alpha_k = \left[\frac{3(1-\zeta)}{4(L_H+\eta)}\right]^{1/2} \frac{\epsilon_H^{1/2}}{\|\mathbf{d}_k\|^{1/2}}.$$

The resulting step $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$ satisfies

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant \bar{c}_{\text{sol}} \epsilon_H^3,$$

Algorithm 4 Inexact Newton-CG without Line Search

Inputs: $\epsilon_g, \epsilon_H > 0$; Parameter $\theta \in (0,1)$; Starting point $\mathbf{x}_0$; upper bound on Hessian norm $U_H > 0$; accuracy parameter $\zeta \in (0, \min\{1, U_H\})$;
for $k = 0, 1, 2, \cdots$ **do**
  if $\|\mathbf{g}_k\| \geqslant \epsilon_g$ **then**
    Call Procedure 1 with $\mathbf{H} = \mathbf{H}_k, M = U_H, \epsilon = \epsilon_H, \mathbf{g} = \mathbf{g}_k$ and accuracy parameter ζ to obtain $\mathbf{d}$ and d_{type};
    if $d_{\text{type}} == NC$ **then**
      $\mathbf{d}_k \leftarrow -\text{sgn}(\mathbf{d}^T\mathbf{g}_k)\frac{|\mathbf{d}^T\mathbf{H}_k\mathbf{d}|}{\|\mathbf{d}\|^2}\frac{\mathbf{d}}{\|\mathbf{d}\|}$;
    else
      $\mathbf{d}_k \leftarrow \mathbf{d}$;
      if $\|\mathbf{d}_k\| \leqslant \epsilon_g/\epsilon_H$ **then**
        Call Procedure 2 with $\mathbf{H} = \mathbf{H}_k, M = U_H, \epsilon = \epsilon_H$ to obtain $\mathbf{v}$ with $\|\mathbf{v}\| = 1$ and $\mathbf{v}^T\mathbf{H}_k\mathbf{v} \leqslant -\epsilon_H/2$ or a certificate that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon_H$;
        if Procedure 2 certifies that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon_H$ **then**
          Terminate and return $\mathbf{x}_k + \mathbf{d}_k$;
        else
          $\mathbf{d}_k \leftarrow -\text{sgn}(\mathbf{v}^T\mathbf{g}_k)|\mathbf{v}^T\mathbf{H}_k\mathbf{v}|\mathbf{v}$ and $d_{\text{type}} \leftarrow$ NC;
        end if
      end if
    end if
  else
    $d_{\text{type}} \leftarrow$ NC;
    Call Procedure 2 with $\mathbf{H} = \mathbf{H}_k, M = U_H, \epsilon = \epsilon_H$ to obtain $\mathbf{v}$ with $\|\mathbf{v}\| = 1$ and $\mathbf{v}^T\mathbf{H}_k\mathbf{v} \leqslant -\epsilon_H/2$ or a certificate that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon_H$;
    if Procedure 2 certifies that $\lambda_{\min}(\mathbf{H}_k) \geqslant -\epsilon$ **then**
      Terminate and return $\mathbf{x}_k$;
    else
      $\mathbf{d}_k \leftarrow -\text{sgn}(\mathbf{v}^T\mathbf{g}_k)|\mathbf{v}^T\mathbf{H}_k\mathbf{v}|\mathbf{v}$;
    end if
  end if

Continued

if $d_{\text{type}} ==$ NC **then**

  Define α_k as in Lemma 2.10, to satisfy $\alpha_k \geqslant \frac{3}{4}\frac{\tilde{\theta}}{L_H+\eta}$ for some $\tilde{\theta} \in ((2-\sqrt{3})^2, 1)$

else

  $\alpha_k = \left[\frac{3(1-\zeta)}{4(L_H+\eta)}\right]^{1/2} \frac{\epsilon_H^{1/2}}{\|\mathbf{d}_k\|^{1/2}}$ (defined in Lemma 2.9)

end if

  $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \alpha_k \mathbf{d}_k$;

end for

where

$$\bar{c}_{\text{sol}} = \frac{\eta}{6}\left[\frac{3(1-\zeta)}{4L_H(L_H+\eta)}\right]^{3/2}.$$

Proof. First, we prove that $\alpha_k \leqslant 1$. We have, using $\epsilon_H = \sqrt{L_H \epsilon_g}$, that

$$\alpha_k^2 = \frac{3(1-\zeta)\epsilon_H}{4(L_H+\eta)\|\mathbf{d}_k\|} \leqslant \frac{3(1-\zeta)\epsilon_H^2}{4(L_H+\eta)\epsilon_g} = \frac{3(1-\zeta)L_H}{4(L_H+\eta)} < 1.$$

If we can show that (2.29) holds, then we obtain the conclusion of the lemma by substituting the formula for α_k into this expression, and using $\|\mathbf{d}_k\| \geqslant \epsilon_g/\epsilon_H$ and $\epsilon_H = \sqrt{L_H \epsilon_g}$.

Suppose for contradiction that condition (2.29) is not satisfied. Then we have

$$\begin{aligned}
-\frac{\eta}{6}\alpha_k^3\|\mathbf{d}_k\|^3 &\leqslant f(\mathbf{x}_k + \alpha_k\mathbf{d}_k) - f(\mathbf{x}_k) \\
&\leqslant \alpha_k \nabla f_k^T \mathbf{d}_k + \frac{\alpha_k^2}{2}\mathbf{d}_k^T \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k + \frac{L_H}{6}\alpha_k^3\|\mathbf{d}_k\|^3 \\
&= \alpha_k \mathbf{g}_k^T \mathbf{d}_k + \frac{\alpha_k^2}{2}\mathbf{d}_k^T \mathbf{H}_k \mathbf{d}_k + \alpha_k(\nabla f_k - \mathbf{g}_k)^T \mathbf{d}_k + \frac{\alpha_k^2}{2}\mathbf{d}_k^T(\nabla^2 f(\mathbf{x}_k) - \mathbf{H}_k)\mathbf{d}_k \\
&\quad + \frac{L_H}{6}\alpha_k^3\|\mathbf{d}_k\|^3 \\
&\leqslant -\frac{\alpha_k}{2}(1-\zeta)\epsilon_H\|\mathbf{d}_k\|^2 + \alpha_k\delta_{g,k}\|\mathbf{d}_k\| + \frac{\alpha_k^2}{2}\delta_H\|\mathbf{d}_k\|^2 + \frac{L_H}{6}\alpha_k^3\|\mathbf{d}_k\|^3 \qquad \text{(from (2.11))} \\
&< \alpha_k\delta_{g,k}\|\mathbf{d}_k\| - \frac{\alpha_k}{2}\|\mathbf{d}_k\|^2\big((1-\zeta)\epsilon_H - \delta_H\big) + \frac{L_H}{6}\alpha_k^3\|\mathbf{d}_k\|^3 \qquad \text{(since } \alpha_k < 1\text{)}.
\end{aligned}$$

By rearrangement, it follows that

$$\frac{L_H+\eta}{6}\alpha_k^2\|\mathbf{d}_k\|^2-\frac{1}{2}\big((1-\zeta)\epsilon_H-\delta_H\big)\|\mathbf{d}_k\|+\delta_{g,k}>0. \tag{2.30}$$

By substituting the definition of α_k and using $\delta_H \leqslant (1-\zeta)\epsilon_H/2$ into the formula above, we have that (2.30) implies

$$\frac{L_H+\eta}{6}\left[\frac{3(1-\zeta)}{4(L_H+\eta)}\right]\frac{\epsilon_H}{\|\mathbf{d}_k\|}\|\mathbf{d}_k\|^2-\frac{(1-\zeta)\epsilon_H}{4}\|\mathbf{d}_k\|+\delta_{g,k}>0$$
$$\Leftrightarrow -\frac{(1-\zeta)}{8}\epsilon_H\|\mathbf{d}_k\|+\delta_{g,k}>0.$$

By using $\delta_{g,k}\leqslant(1-\zeta)\max\left(\epsilon_g,\min(\epsilon_H\|\mathbf{d}_k\|,\|\mathbf{g}_k\|,\|\mathbf{g}_{k+1}\|)\right)/8$, this inequality implies that

$$-\epsilon_H\|\mathbf{d}_k\|+\max\left(\epsilon_g,\min(\epsilon_H\|\mathbf{d}_k\|,\|\mathbf{g}_k\|,\|\mathbf{g}_{k+1}\|)\right)>0. \tag{2.31}$$

If $\epsilon_g>\min(\epsilon_H\|\mathbf{d}_k\|,\|\mathbf{g}_k\|,\|\mathbf{g}_{k+1}\|)$, since $\epsilon_H=\sqrt{L_H\epsilon_g}$, we have from (2.31) that

$$-\sqrt{L_H\epsilon_g}\|\mathbf{d}_k\|+\epsilon_g>0\Rightarrow\|\mathbf{d}_k\|<\sqrt{\epsilon_g/L_H},$$

which contradicts our assumption $\|\mathbf{d}_k\|\geqslant\sqrt{\epsilon_g/L_H}=\epsilon_g/\epsilon_H$. Alternatively, if we assume that $\epsilon_g\leqslant\min(\epsilon_H\|\mathbf{d}_k\|,\|\mathbf{g}_k\|,\|\mathbf{g}_{k+1}\|)$, then from (2.31), it follows that

$$0<-\epsilon_H\|\mathbf{d}_k\|+\min(\epsilon_H\|\mathbf{d}_k\|,\|\mathbf{g}_k\|,\|\mathbf{g}_{k+1}\|)\leqslant-\epsilon_H\|\mathbf{d}_k\|+\epsilon_H\|\mathbf{d}_k\|=0,$$

which is again a contradiction. Hence, our chosen value of α_k must satisfy (2.29), completing the proof. □

Next, let us deal with the case when $d_{\text{type}}=\text{NC}$, which can be considered as a fixed-step alternative to Lemma 2.6.

LEMMA 2.10 Suppose that Assumption 1 is satisfied and that Condition 2.4 holds for all k. Suppose that at iteration k of Algorithm 4, we have $\|\mathbf{g}_k\|>\epsilon_g$, so that Procedure 1 is called. When Procedure 1 outputs a direction $\mathbf{d}_k$ with $d_{\text{type}}=\text{NC}$, we can choose the pre-defined step size

$$\alpha_k=\left(\frac{(\|\mathbf{d}_k\|-\delta_H)/2+\sqrt{((\|\mathbf{d}_k\|-\delta_H)/2)^2-4(L_H+\eta)\delta_{g,k}/6}}{(L_H+\eta)\|\mathbf{d}_k\|/3}\right)\tilde{\theta},$$

where $\tilde{\theta}$ is a parameter satisfying $(2-\sqrt{3})^2 < \tilde{\theta} < 1$. The resulting step $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$ satisfies $f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \geqslant \bar{c}_{\rm nc}\epsilon_H^3$, where

$$\bar{c}_{\rm nc} := \frac{\eta}{6}\left[\frac{3\tilde{\theta}}{4(L_H+\eta)}\right]^3.$$

Proof. We start by noting that under the assumptions of the lemma, we have

$$\mathbf{d}_k^T \mathbf{H}_k \mathbf{d}_k \leqslant -\epsilon_H \|\mathbf{d}_k\|^2, \quad \|\mathbf{d}_k\| \geqslant \epsilon_H, \tag{2.32}$$

We replace the lower bound on $\|\mathbf{d}_k\|$ by the weaker bound $\|\mathbf{d}_k\| \geqslant \frac{1}{2}\epsilon_H$ (so that we can reuse our results in the next lemma), to obtain

$$\|\mathbf{d}_k\| \geqslant \frac{1}{2}\epsilon_H, \quad \delta_H \leqslant \frac{1}{4}\epsilon_H \leqslant \frac{1}{2}\|\mathbf{d}_k\| \text{ and so } \|\mathbf{d}_k\| - \delta_H \geqslant \frac{1}{2}\|\mathbf{d}_k\| \geqslant \frac{1}{4}\epsilon_H. \tag{2.33}$$

Note too that $\mathbf{d}_k^T \mathbf{g}_k \leqslant 0$ by design, so that from Definition 2.1 of $\delta_{g,k}$, we have

$$\mathbf{d}_k^T \nabla f(\mathbf{x}_k) \leqslant \mathbf{d}_k^T \mathbf{g}_k + \|\mathbf{d}_k\|\|\nabla f(\mathbf{x}_k) - \mathbf{g}_k\| \leqslant \delta_{g,k}\|\mathbf{d}_k\|. \tag{2.34}$$

We therefore have

$$\begin{aligned} f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) - \mathbf{f}(\mathbf{x}_k) &\leqslant \alpha_k \nabla f(\mathbf{x}_k)^T \mathbf{d}_k + \frac{\alpha_k^2}{2}\mathbf{d}_k^T \nabla^2 f(\mathbf{x}_k)\mathbf{d}_k + \frac{L_H}{6}\alpha_k^3\|\mathbf{d}_k\|^3 \\ &\leqslant \alpha_k \delta_{g,k}\|\mathbf{d}_k\| - \frac{\alpha_k^2}{2}\|\mathbf{d}_k\|^3 + \frac{\alpha_k^2}{2}\delta_H\|\mathbf{d}_k\|^2 + \frac{L_H}{6}\alpha_k^3\|\mathbf{d}_k\|^3. \qquad \text{(from (2.20))} \end{aligned}$$

Thus condition (2.29) will be satisfied provided that

$$\alpha_k \delta_{g,k}\|\mathbf{d}_k\| - \frac{\alpha_k^2}{2}\|\mathbf{d}_k\|^3 + \frac{\alpha_k^2}{2}\delta_H\|\mathbf{d}_k\|^2 + \frac{L_H}{6}\alpha_k^3\|\mathbf{d}_k\|^3 \leqslant -\frac{\eta}{6}\alpha_k^3\|\mathbf{d}_k\|^3.$$

By rearranging and dividing by $\alpha_k\|\mathbf{d}_k\|$, we find that α_k satisfies (2.29) provided that the following quadratic inequality in α_k is satisfied:

$$\left(\frac{(L_H+\eta)\|\mathbf{d}_k\|^2}{6}\right)\alpha_k^2 - \left(\frac{\|\mathbf{d}_k\|(\|\mathbf{d}_k\| - \delta_H)}{2}\right)\alpha_k + \delta_{g,k} \leqslant 0. \tag{2.35}$$

In fact, this inequality is satisfied provided that $\alpha_k \in [\beta_2, \beta_1]$, where

$$\beta_1 := \frac{(\|\mathbf{d}_k\| - \delta_H)/2 + \sqrt{((\|\mathbf{d}_k\| - \delta_H)/2)^2 - 4(L_H + \eta)\delta_{g,k}/6}}{(L_H + \eta)\|\mathbf{d}_k\|/3},$$
$$\beta_2 := \frac{(\|\mathbf{d}_k\| - \delta_H)/2 - \sqrt{((\|\mathbf{d}_k\| - \delta_H)/2)^2 - 4(L_H + \eta)\delta_{g,k}/6}}{(L_H + \eta)\|\mathbf{d}_k\|/3}.$$

To verify that the quantity under the square root is positive, we use (2.33) to write

$$\left(\frac{\|\mathbf{d}_k\| - \delta_H}{2}\right)^2 - 4\frac{(L_H + \eta)}{6}\delta_{g,k} \geqslant \frac{1}{16}\|\mathbf{d}_k\|^2 - \frac{2(L_H + \eta)}{3}\delta_{g,k}$$
$$\geqslant \frac{1}{64}\epsilon_H^2 - \frac{2(L_H + \eta)}{3}\delta_{g,k} > 0,$$

where the last inequality follows from Condition 2.4, since

$$\delta_{g,k} \leqslant \frac{3}{2 \times 65}\frac{\epsilon_H^2}{L_H + \eta} < \frac{3}{128}\frac{\epsilon_H^2}{L_H + \eta}.$$

(Note that $0 < \beta_2 < \beta_1$.)

Next, we show that our choice of α_k, which equals $\tilde{\theta}\beta_1$, lies in the interval (β_2, β_1). First, we have $\alpha_k = \tilde{\theta}\beta_1 < \beta_1$ since $\tilde{\theta} < 1$. Secondly, proving $\alpha_k > \beta_2$ is equivalent to showing that $\tilde{\theta} > \beta_2/\beta_1$. Defining

$$z := \frac{\|\mathbf{d}_k\| - \delta_H}{2}, \quad c := \frac{2}{3}(L_H + \eta)\delta_{g,k},$$

we see that

$$\beta_1 = \frac{z + \sqrt{z^2 - c}}{(L_H + \eta)\|\mathbf{d}_k\|/3}, \quad \beta_2 = \frac{z - \sqrt{z^2 - c}}{(L_H + \eta)\|\mathbf{d}_k\|/3},$$

so that the required condition is

$$\tilde{\theta} > \beta_2/\beta_1 = \frac{z - \sqrt{z^2 - c}}{z + \sqrt{z^2 - c}} = \frac{(z - \sqrt{z^2 - c})^2}{c}.$$

We have from (2.33) and Condition 2.5 that

$$z^2 = \left(\frac{\|\mathbf{d}_k\| - \delta_H}{2}\right)^2 \geqslant \frac{\epsilon_H^2}{64} > \frac{\epsilon_H^2}{65} \geqslant \frac{8}{3}(L_H + \eta)\delta_{g,k} = 4c.$$

Since $z - \sqrt{z^2 - c}$ is a decreasing function of z for all $z^2 > c > 0$, we have by using $z^2 > 4c$ that

$$\frac{\beta_2}{\beta_1} = \frac{(z - \sqrt{z^2 - c})^2}{c} < \frac{(2\sqrt{c} - \sqrt{4c - c})^2}{c} = (2 - \sqrt{3})^2 < \tilde{\theta}.$$

We have therefore proved that $\alpha_k \in [\beta_2, \beta_1]$, so that α_k satisfies (2.29).

From (2.33), we have

$$\begin{aligned}\alpha_k = \tilde{\theta}\beta_1 &= \tilde{\theta}\frac{(\|\mathbf{d}_k\| - \delta_H)/2 + \sqrt{((\|\mathbf{d}_k\| - \delta_H)/2)^2 - 4(L_H + \eta)\delta_{g,k}/6}}{(L_H + \eta)\|\mathbf{d}_k\|/3} \\ &\geqslant \tilde{\theta}\frac{\|\mathbf{d}_k\|/4}{(L_H + \eta)\|\mathbf{d}_k\|/3} = \frac{3}{4}\frac{\tilde{\theta}}{L_H + \eta}. \end{aligned} \tag{2.36}$$

The final claim of the theorem is obtained by substituting this lower bound on α_k into (2.29), and using $\|\mathbf{d}_k\| \geqslant \epsilon_H$. □

The next lemma shows that when $d_{\text{type}} = \text{NC}$ is obtained from Procedure 2, the same fixed step size as in Lemma 2.10 can be used, with the same lower bound on improvement in f.

LEMMA 2.11 Suppose that Assumption 1 is satisfied and that Condition 2.4 holds for all k. Suppose that at iteration k of Algorithm 4, the step $\mathbf{d}_k$ is of negative curvature type, obtained from Procedure 2. Then when we define α_k as in Lemma 2.10, we obtain

$$f(\mathbf{x}_k) - f(\mathbf{x}_k + \alpha_k\mathbf{d}_k) \geqslant \frac{1}{8}\bar{c}_{\text{nc}}\epsilon_H^3, \tag{2.37}$$

where $\bar{c}_{\text{nc}}$ is defined in Lemma 2.10.

Proof. Note that for $\mathbf{d}_k$ obtained from Procedure 2, we have

$$\mathbf{d}_k^T\mathbf{H}\mathbf{d}_k \leqslant -\frac{1}{2}\epsilon_H\|\mathbf{d}_k\|^2, \quad \|\mathbf{d}_k\| \geqslant \frac{1}{2}\epsilon_g.$$

Since the bulk of the proof of Lemma 2.10 uses only the latter lower bound on $\|\mathbf{d}_k\|$, we can use this proof to derive the same lower bound (2.36) on α_k. The result follows by substituting this lower bound together with $\|\mathbf{d}_k\| \geqslant \epsilon_H/2$ into (2.29). □

Using Lemma 2.9 to 2.8, we are now ready to give the iteration complexity of Algorithm 4.

THEOREM 2.5 Suppose that Assumption 1 is satisfied and that Condition 2.4 holds for all k. For a given $\epsilon > 0$, let $\epsilon_H = \sqrt{L_H\epsilon}, \epsilon_g = \epsilon$. Define

$$\bar{K}_2 := 2\left\lceil \frac{f(\mathbf{x}_0) - f_{\text{low}}}{\min\{\bar{c}_{\text{sol}}, \bar{c}_{\text{nc}}/8\}L_H^{3/2}}\epsilon^{-3/2}\right\rceil + 3, \tag{2.38}$$

where $\bar{c}_{\text{sol}}$ and $\bar{c}_{\text{nc}}$ are defined in Lemma 2.9 and Lemma 2.10, respectively. Then Algorithm 4 terminates in at most $\bar{K}_2$ iterations at a point $\mathbf{x}$ satisfying $\|\nabla f(\mathbf{x})\| \lesssim \epsilon$. Moreover, with probability at least $(1-\delta)^{\bar{K}_2}$,

the point returned by Algorithm 4 also satisfies the approximate second-order condition $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \gtrsim -\sqrt{L_H \epsilon}$. Here again, $\lesssim$ and $\gtrsim$ denote that the corresponding inequality holds up to a certain constant that is independent of ϵ and L_H.

Proof. The proof tracks that of Theorem 2.3 closely, so we omit much of the detail and discussion.

For contradiction, we assume that Algorithm 4 runs for at least K steps, where $K > \bar{K}_2$. We partition the set of iteration indices $\{1, 2, \ldots, K\}$ into the same sets $\mathcal{K}_1, \ldots, \mathcal{K}_5$ as in the proof of Theorem 2.3. Considering each of these sets in turn, we have the following.

Case 1: $k \in \mathcal{K}_1$. Either Algorithm 4 terminates (which happens at most once for $k \in \mathcal{K}_1$) or we achieve a reduction in f of at least $\frac{1}{8}\bar{c}_{\mathrm{nc}}\epsilon_H^3 = \frac{1}{8}\bar{c}_{\mathrm{nc}} L_H^{3/2}\epsilon^{3/2}$ (Lemma 2.11).

Cases 2 and 3: $k \in \mathcal{K}_2 \cup \mathcal{K}_3$. f is reduced by at least $\bar{c}_{\mathrm{sol}} L_H^{3/2}\epsilon^{3/2}$ (Lemma 2.9). Note that, unlike Case 2 in Theorem 2.3, we can indeed guarantee a positive decrease in f for all $k \in \mathcal{K}_2$. This is because, unlike Lemma 2.5, the decrease obtained from Lemma 2.9 does not depend on $\|\mathbf{g}_{k+1}\|$.

Case 4: $k \in \mathcal{K}_4$. The algorithm terminates, so we must have $|\mathcal{K}_4| \leqslant 1$.

Case 5: $k \in \mathcal{K}_5$. Either the algorithm terminates, or we achieve a reduction of at least $\bar{c}_{\mathrm{nc}} L_H^{3/2}\epsilon^{3/2}$ (Lemma 2.10).

Reasoning as in the proof of Theorem 2.3, we have that

$$f(\mathbf{x}_0) - f_{\mathrm{low}} \geqslant (|\mathcal{K}_1| - 1)\tfrac{1}{8}\bar{c}_{\mathrm{nc}} L_H^{3/2}\epsilon^{3/2} + (|\mathcal{K}_2| + |\mathcal{K}_3|)\bar{c}_{\mathrm{sol}} L_H^{3/2}\epsilon^{3/2} + (|\mathcal{K}_5| - 1)\bar{c}_{\mathrm{nc}} L_H^{3/2}\epsilon^{3/2},$$

from which we obtain

$$|\mathcal{K}_1| + |\mathcal{K}_5| - 2 \leqslant \frac{f(\mathbf{x}_0) - f_{\mathrm{low}}}{\frac{1}{8}\bar{c}_{\mathrm{nc}} L_H^{3/2}}\epsilon^{-3/2},$$

$$|\mathcal{K}_2| + |\mathcal{K}_3| \leqslant \frac{f(\mathbf{x}_0) - f_{\mathrm{low}}}{\bar{c}_{\mathrm{sol}} L_H^{3/2}}\epsilon^{-3/2}.$$

By using these bounds along with $|\mathcal{K}_4| \leqslant 1$, we obtain

$$K \leqslant \sum_{i=1}^{5} |\mathcal{K}_i| \leqslant 2\frac{f(\mathbf{x}_0) - f_{\mathrm{low}}}{\min(\bar{c}_{\mathrm{nc}}/8, \bar{c}_{\mathrm{sol}}) L_H^{3/2}}\epsilon^{-3/2} + 3,$$

which contradicts our assumption that $K > \bar{K}_2$.

The proof of the remaining claim, concerning the approximate second-order condition, is identical to the corresponding section in the proof of Theorem 2.3. □

Note that the worst-case iteration complexity of Algorithm 4 has the same dependence on ϵ as Algorithm 3, despite the function evaluation no longer being required. The terms in the bound that do not depend on ϵ are, however, generally worse for Algorithm 4.

We conclude with a discussion of Conditions 2.2 and 2.4. These conditions allow for the accuracy of $\mathbf{g}_k$ to be chosen adaptively, depending on problem-dependent constants, algorithmic parameters, the desired solution tolerances ϵ_g and ϵ_H, and the quantities $\|\mathbf{d}_k\|$, $\|g_{k+1}\|$ and $\|\mathbf{g}_k\|$. The quantity $\|\mathbf{g}_k\|$ is easy to evaluate (since, after all, $\mathbf{g}_k$ is the quantity actually calculated). However, the dependence on the quantities $\|\mathbf{d}_k\|$ and $\|\mathbf{g}_{k+1}\|$ is more problematic, since $\mathbf{g}_k$ is needed to compute both $\mathbf{d}_k$ and $\mathbf{g}_{k+1}$. Thus, the bounds on $\delta_{g,k}$ in Conditions 2.2 and 2.4 can be checked only 'in retrospect', not enforced as an *a priori* condition. We can deal with this issue by checking the bound on $\delta_{g,k}$ after the step to $\mathbf{x}_{k+1}$ has been taken. if it fails to be satisfied, we can improve the accuracy of $\mathbf{g}_k$ and re-do iteration k. If we halve $\delta_{g,k}$ each time the step is recomputed, the number of recomputations is at worst a multiple of $\log \epsilon_g$ (since the bound on $\delta_{g,k}$ in both conditions is at least $(1-\zeta)\epsilon_g/8$), so our complexity bounds are not affected significantly. We choose to elide this fairly uninteresting issue in our analysis, and simply assume for simplicity that the relevant bound on $\delta_{g,k}$ holds at each iteration.

2.5 *Evaluation complexity of Algorithm 4 for finite-sum problems*

When f has finite-sum form (1.2) for $n \gg 1$, we consider subsampling schemes for estimating $\mathbf{g}_k$ and $\mathbf{H}_k$, as in Roosta & Mahoney (2019); Xu *et al.* (2020a). We can define the subsampled quantities as follows

$$\mathbf{g} \triangleq \frac{1}{\left|\mathcal{S}_g\right|} \sum_{i \in \mathcal{S}_g} \nabla f_i(\mathbf{x}) \quad \text{and} \quad \mathbf{H} \triangleq \frac{1}{\left|\mathcal{S}_H\right|} \sum_{i \in \mathcal{S}_H} \nabla^2 f_i(\mathbf{x}), \tag{2.39}$$

where $\mathcal{S}_g, \mathcal{S}_H \subset \{1, \cdots, n\}$ are the subsample batches for the estimates of the gradient and Hessian, respectively. In Roosta & Mahoney (2019, Lemma 1 and 2) and Xu *et al.* (2020a, Lemma 16), it is shown that with a uniform sampling strategy, the following lemma can be proved.

LEMMA 2.12 (Sampling complexity, Roosta & Mahoney, 2019; Xu *et al.*, 2020a). Suppose that Assumption 1 is satisfied, and let $\bar{\delta} \in (0,1)$ be given. Suppose that at iteration k of Algorithm 4, $\delta_{g,k}$ and δ_H are as defined in Condition 2.4. Also, let $0 < K_g, K_H < \infty$ be such that $\left\|\nabla f_i(\mathbf{x})\right\| \leqslant K_g$ and $\left\|\nabla^2 f_i(\mathbf{x})\right\| \leqslant K_H$ for all $\mathbf{x}$ belonging to the set defined in Assumption 1. For $\mathbf{g}_k$ and $\mathbf{H}_k$ defined as in (2.39) with $\mathbf{x} = \mathbf{x}_k$, and subsample sets $\mathcal{S}_g = \mathcal{S}_{g,k}$ and $\mathcal{S}_H$, satisfying

$$\left|\mathcal{S}_{g,k}\right| \geqslant \frac{16K_g^2}{\delta_{g,k}^2} \log \frac{1}{\bar{\delta}} \quad \text{and} \quad \left|\mathcal{S}_H\right| \geqslant \frac{16K_H^2}{\delta_H^2} \log \frac{2d}{\bar{\delta}},$$

we have with probability at least $1 - \bar{\delta}$ that Condition 2.4 holds for the given values of $\delta_{g,k}$ and δ_H.

For the choices of ϵ_g and ϵ_H being used in this section, and assuming that $\delta_{g,k}$ and δ_H are set to their upper bounds in Condition 2.4, we can derive a uniform condition on the required subsample sizes.

LEMMA 2.13 Suppose the conditions of Lemma 2.12 holds, and that for some $\epsilon > 0$, we set $\epsilon_H = \sqrt{L_H \epsilon}$ and $\epsilon_g = \epsilon$. Suppose that at some iteration k, $\delta_{g,k}$ and δ_H are set to their upper bounds in Condition 2.4. Then we have that $\delta_{g,k} \geqslant \bar{\delta}_g$ for all k and $\delta_H = \bar{\delta}_H$, where

$$\bar{\delta}_g = \frac{1-\zeta}{8} \min\left(\frac{3L_H\epsilon}{65(L_H+\eta)}, \epsilon\right) = \mathcal{O}(\epsilon), \quad \bar{\delta}_H = \left(\frac{1-\zeta}{4}\right)\sqrt{L_H\epsilon} = \mathcal{O}(\epsilon^{1/2}). \tag{2.40}$$

Moreover, when $\mathbf{g}_k$ and $\mathbf{H}_k$ are estimated from (2.39) with $\mathbf{x} = \mathbf{x}_k$ and subsample sets $\mathcal{S}_g = \mathcal{S}_{g,k}$ and $\mathcal{S}_H$, satisfying

$$\left|\mathcal{S}_g\right| \geqslant \frac{16K_g^2}{\bar{\delta}_g^2}\log\frac{1}{\bar{\delta}} = \mathcal{O}(\epsilon^{-2}), \quad \left|\mathcal{S}_H\right| \geqslant \frac{16K_H^2}{\bar{\delta}_H^2}\log\frac{2d}{\bar{\delta}} = \mathcal{O}(\epsilon^{-1}),$$

then Condition 2.4 is satisfied at iteration k with probability at least $1 - \bar{\delta}$.

Proof. The right-hand side of the bound on $\delta_{g,k}$ in Condition 2.4 is bounded below by

$$\frac{1-\zeta}{8}\min\left(\frac{3\epsilon_H^2}{65(L_H+\eta)}, \epsilon_g\right) = \frac{1-\zeta}{8}\min\left(\frac{3L_H\epsilon}{65(L_H+\eta)}, \epsilon\right) = \bar{\delta}_g = \mathcal{O}(\epsilon),$$

as claimed. The claims concerning $\bar{\delta}_H$ are immediate. □

By combining Lemma 2.13 with Theorem 2.5, we can obtain an oracle complexity result in which the oracle is either an evaluation of a gradient ∇f_i for some $i = 1, 2, \ldots, n$ or a Hessian-vector product of the form $\nabla^2 f_i(\mathbf{x})\mathbf{v}$, for some $i = 1, 2, \ldots, n$ and some $\mathbf{x}, \mathbf{v} \in \mathbb{R}^d$. The result is complicated by the fact that there is a probability of failure to satisfy Condition 2.4 at each k, to go along with the possible failure, noted in the previous section, to detect negative curvature when Procedure 2 is invoked. For our result below, we consider the case in which failure to satisfy Condition 2.4 never occurs at any iteration, Since there are at most $\bar{K}_2$ iterations, this case occurs with probability at least $(1-\bar{\delta})^{\bar{K}_2}$.

Corollary 2.2 (Evaluation complexity of Algorithm 4 for finite-sum problem (1.2)). Suppose that Assumption 1 is satisfied. Let $\bar{\delta} \in (0,1)$ be given, and suppose that at each iteration k, $\mathbf{g}_k$ and $\mathbf{H}_k$ are obtained from (2.39), with $\mathcal{S}_g = \mathcal{S}_{g,k}$ and $\mathcal{S}_H$ satisfying the lower bounds in Lemma 2.12, where $\delta_{g,k} \geqslant \bar{\delta}_g$ and $\delta_H \geqslant \bar{\delta}_H$, with $\bar{\delta}_g$ and $\bar{\delta}_H$ defined in (2.40). For a given $\epsilon > 0$, let $\epsilon_H = \sqrt{L_H\epsilon}, \epsilon_g = \epsilon$. Let $\bar{K}_2$ be defined as in (2.38). Then with probability at least $(1-\bar{\delta})^{\bar{K}_2}(1-\delta)^{\bar{K}_2}$, Algorithm 4 terminates in at most $\bar{K}_2$ iterations at a point $\mathbf{x}$ satisfying $\|\nabla f(\mathbf{x})\| \lesssim \epsilon$ and $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \gtrsim -\sqrt{L_H\epsilon}$. Again, $\lesssim$ and $\gtrsim$ denote that the corresponding inequality holds up to a certain constant that is independent of ϵ and L_H. Moreover, the total number of oracle calls is bounded by

$$\begin{aligned}
&\underbrace{\left(2\left\lceil\frac{(f(\mathbf{x}_0)-f_{\text{low}})}{\min\{\bar{c}_{\text{sol}},\bar{c}_{\text{nc}}/8\}}\epsilon^{-3/2}\right\rceil+3\right)}_{\bar{K}_2}\cdot\left(\underbrace{\frac{16K_g^2}{\bar{\delta}_g^2}\log\frac{1}{\bar{\delta}}}_{\text{Gradient Sampling}}+\underbrace{\frac{16K_H^2}{\bar{\delta}_H^2}\log\frac{2d}{\bar{\delta}}}_{\text{Hessian Sampling}}\cdot\left(\underbrace{\tilde{\mathcal{O}}(\epsilon^{-1/4})}_{\textit{Procedure 1}}+\underbrace{\mathcal{O}(\epsilon^{-1/4})}_{\textit{Procedure 2}}\right)\right)\\
&= \mathcal{O}(\epsilon^{-3/2})\cdot(\mathcal{O}(\epsilon^{-2})+\tilde{\mathcal{O}}(\epsilon^{-1}\times\epsilon^{-1/4}))\\
&= \mathcal{O}(\epsilon^{-7/2}).
\end{aligned}$$

As mentioned earlier, Algorithm 4 requires knowledge of an upper bound of the Lipschitz constant L_H of the Hessian matrix. In addition, not only do the sample complexities derived in Corollary 2.2 depend on L_H through the use of $\bar{\delta}_g$ and $\bar{\delta}_H$ as in (2.40), but they also require upper estimates of K_g and K_H, which may be unavailable for many nonconvex problems. Fortunately, for many nonconvex

objectives of interest in machine learning and statistical analysis, we can readily obtain reasonable estimates of these quantities. Table 1 provides estimates on L_H for some examples of such objectives. See Table 2 for upper bounds on K_g and K_H for such problems. Equipped with these estimates, we can give a more refined complexity analysis tailored for the problems in Tables 1 and 2. Indeed, since for the constants $\bar{c}_{\text{sol}}$ and $\bar{c}_{\text{nc}}$ in Lemmas 2.10 and 2.9, we have $\bar{c}_{\text{sol}} \in \Omega(1/L_H^3)$, $\bar{c}_{\text{nc}} \in \Omega(1/L_H^3)$, from Tables 1 and 2, it follows that the total number of oracle calls for these problems is at most

$$\tilde{\mathcal{O}}\left[\left(\left(\max_i \|\mathbf{a}_i\|^9\right)(f(\mathbf{x}_0)-f_{\text{low}})\epsilon^{-3/2}\right)\right]\cdot\left(\tilde{\mathcal{O}}\left(\left(\max_i \|\mathbf{a}_i\|^2\right)\epsilon^{-2}\right)+\tilde{\mathcal{O}}\left(\left(\max_i \|\mathbf{a}_i\|\right)\epsilon^{-5/4}\right)\right)$$
$$=\tilde{\mathcal{O}}\left(\epsilon^{-7/2}(f(\mathbf{x}_0)-f_{\text{low}})\max_i\left\{1,\|\mathbf{a}_i\|\right\}^{11}\right),$$

where for simplicity we have assumed $|b_i| \leqslant 1$ as in binary classification problems.

3. Numerical evaluation

In this section, we evaluate the performance of Algorithms 3 and 4 on three model problems in the form of finite-sum minimization: nonlinear least squares (NLS), multilayer perceptron (MLP) and variational autoencoder (VAE). Our aim here is to illustrate the efficiency gained from gradient and Hessian approximations, as compared with the exact counterpart in Royer *et al.* (2020). More specifically, in our numerical examples, we consider the following algorithms.

- `Full NTCG`: Newton Method with Capped-CG solver with full gradient and Hessian evaluations, as developed in Royer *et al.* (2020).
- `SubH NTCG` (**this work**): Variant of Royer *et al.* (2020) where Hessian is approximated. We consider this setting as an intermediary between the full algorithm and those where both the gradient and the Hessian are approximated. Sample sizes for approximating Hessian for experiments using NLS, MLP and VAE, are $0.01n$, $0.02n$ and $0.02n$, respectively.
- `Inexact NTCG Full-Eval` (**this work**): Newton Method with Capped-CG solver with back-tracking line-search where both the gradient and the Hessian are approximated. To perform the backtracking line search, we employ the full dataset to evaluate the objective function. The sample size for estimating the gradient is adaptively calculated as follows: if $\|\mathbf{g}_t\| \geqslant 1.2\|\mathbf{g}_{t-1}\|$ or $\|\mathbf{g}_t\| \leqslant \|\mathbf{g}_{t-1}\|/1.2$, then the sample size is decreased or increased, respectively, by a factor of 1.2. Otherwise, we maintain the same sample size as the previous iteration. The initial sample size to approximate the gradient for the experiments of Section 3.1 is set to $0.05n$, while for the experiments of Sections 3.2 and 3.3, we use an initial sample size of 10,000. The sample size for approximating the Hessian is set the same as that in `SubH NTCG`.
- `Inexact NTCG Fixed` (**this work**): Newton Method with Capped-CG solver, using approximations of both the gradient and the Hessian and fixed step sizes. The step sizes are predefined as follows: for NLS experiments, we use $\alpha_k = 0.04$ for $d_{\text{type}} = \text{NC}$ and $\alpha_k = 0.2$ for $d_{\text{type}} = \text{SOL}$, while for simulations on MLP/VAE models, we consider $\alpha_k = 0.1$ for $d_{\text{type}} = \text{NC}$ and $\alpha_k = \sqrt{0.1}$ for $d_{\text{type}} = \text{SOL}$. The gradient and Hessian approximations are done as in the previous two variants.

- `Inexact NTCG Sub-Eval`: This method is almost identical to `Inexact NTCG Full-Eval`, however, the backtracking line search is performed on estimates of the objective function using the same samples as the ones used in gradient approximation. Of course, our theoretical analysis does not immediately support this variant. However, we have found this strategy to be highly effective in practice, and we intend to theoretically investigate it in future work.

In all of our experiments, we run each stochastic method five times (starting from the same initial point), and plot the average run (solid line) and 1-standard deviation band (shaded regions). To avoid cluttering the plots, we only show the upper deviation from the average, since the lower deviation band is almost identical on all of our experiments.

We note that the step size implied by Algorithm 4 is very pessimistic, and hence small. This is a byproduct of our worst-case analysis, which comprises of descent obtained from a sequence of conservative steps. Requiring small step-lengths to provide a convergence guarantee is perhaps the main drawback for the worst-case style of analysis, which is almost ubiquitous within the optimization literature, e.g., fixed step size of length $1/L_g$ for gradient descent on smooth unconstrained problems. Our numerical example shows that much larger step sizes than those prescribed by Algorithm 4 can be employed in practice. We suspect this to be the case for most practical applications.

Although in Algorithms 3 and 4, the case where $\|\mathbf{d}_k\|$ is small (relative to the ratio ϵ_g/ϵ_H) is crucial in obtaining theoretical guarantees, in all of our simulations, we have found that performing line search directly with such small $\mathbf{d}_k$ and without resorting to Procedure 2 in fact yields reasonable progress. In this light, in all of our implementations, we have made the practical decision to omit Lines 9-16 of Algorithms 3 and 4.

Similar to Xu *et al.* (2020b); Yao *et al.* (2020), the performance of all the algorithms is measured by tallying the total *number of propagations*, that is, the number of oracle calls of function, gradient and Hessian-vector products. This is so, since comparing algorithms in terms of 'wall-clock' time can be highly affected by their particular implementation details, as well as system specifications. In contrast, counting the number of oracle calls, as an implementation and system independent unit of complexity, is most appropriate and fair. More specifically, after computing $f_i(\mathbf{x})$, which accounts for one oracle call, computing the corresponding gradient $\nabla f_i(\mathbf{x})$ is equivalent to one additional function evaluation, i.e., two oracle calls are needed to compute $\nabla f_i(\mathbf{x})$. Our implementations are Hessian-free, i.e., we merely require Hessian-vector products instead of using the explicit Hessian. For this, each Hessian-vector product $\nabla^2 f_i(\mathbf{x})\boldsymbol{v}$ amounts to two additional function evaluations, as compared with gradient evaluation, i.e., four oracle calls are used to evaluate $\nabla^2 f_i(\mathbf{x})\boldsymbol{v}$.

3.1 *Nonlinear least squares*

We first consider the simple, yet illustrative, nonlinear least squares problems arising from the task of binary classification with squared loss.[1] Given training data $\{\mathbf{a}_i, b_i\}_{i=1}^n$, where $\mathbf{a}_i \in \mathbb{R}^d, b_i \in \{0, 1\}$, we solve the empirical risk minimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n}\Big(b_i - \phi\big(\langle \mathbf{a}_i, \mathbf{x}\rangle\big)\Big)^2,$$

[1] Logistic loss, the 'standard' loss used in this task, leads to a convex objective. We use squared loss to obtain a nonconvex objective.

TABLE 3 *Datasets used for NLP experiments*

DATA	n	d
`covertype`	464,810	54
`ijcnn1`	49,990	22

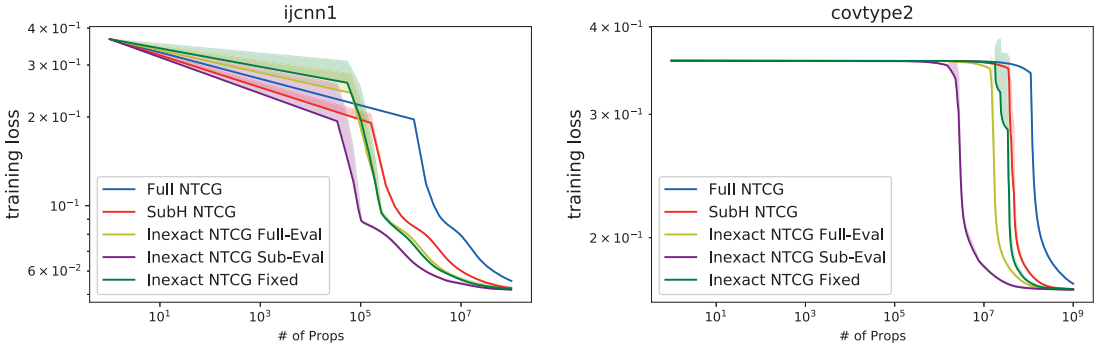

FIG. 1. Comparison between all variants of `NTCG` on `ijcnn1` and `covertype` datasets.

where $\phi(z)$ is the sigmoid function: $\phi(z) = 1/(1 + e^{-z})$. Datasets are taken from `LIBSVM` library (Chang & Lin, 2011); see Table 3 for details. We use the same setup as in Yao *et al.* (2020).

The comparison between different NTCG algorithms is shown in Fig. 1. It is clear that, for a given value of the loss, all inexact variants in the `Inexact NTCG` family converge faster, i.e., with fewer oracle calls. Clearly, lower per-iteration cost of `Inexact NTCG Fixed` comes at the cost of slower overall convergence as compared with `Inexact NTCG Sub-Eval`. This is mainly because the step size obtained as part of the line-search procedure can generally result in a better decrease in function value. For this problem we could refer to Table 1 and explicitly compute the fixed step size prescribed by Algorithm 4. As mentioned earlier, the resulting step size is overly conservative. Our simulations show that much larger step sizes yield convergent algorithms. In this light, our fixed step sizes are chosen without regard to the value prescribed in Algorithm 4, but are based rather on numerical experience.

3.2 *Multilayer perceptron*

Here, we consider a slightly more complex setting than simple NLS, and evaluate the performance of Algorithms 3 and 4 on several MLPs in the context of the image classification problem. For our experiments here, we will make use of the `MNIST` dataset, which is also available from `LIBSVM` library (Chang & Lin, 2011). We consider three MLPs with one hidden layer, involving 16, 128 and 1024 neurons, respectively. All MLPs contain one output layer to determine the assigned class of the input image. The intermediate activation is chosen as the SoftPlus function (Glorot *et al.*, 2011), which amounts to a smooth optimization problem. Table 4 summarizes the total dimensions, in terms of n and d, of the resulting optimization problems.

Figure 2 depicts the performance of all variants of NTCG that we consider in this paper. As can be seen, for all cases, our `Inexact NTCG Full-Eval` and `Inexact NTCG Sub-Eval` have the fastest convergence rate and achieve lower training loss as compared to alternatives.

TABLE 4 *The problem size for various MLPs*

Hidden layer size	n	d
16	60,000	12,704
128	60,000	101,632
1,024	60,000	813,056

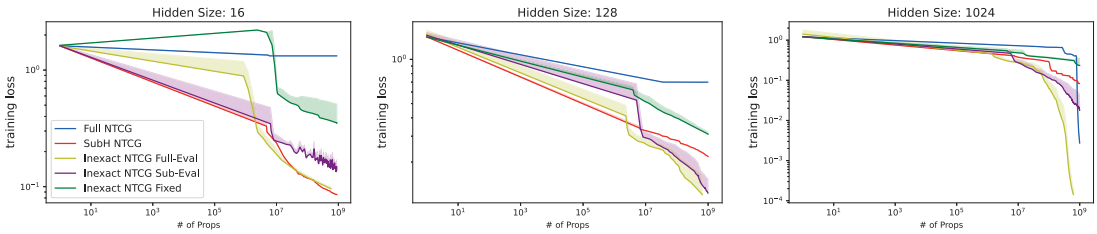

FIG. 2. Comparison between all variants of NTCG on several MLPs with different hidden-layer sizes: 16 (left), 128 (middle) and 1024 (right).

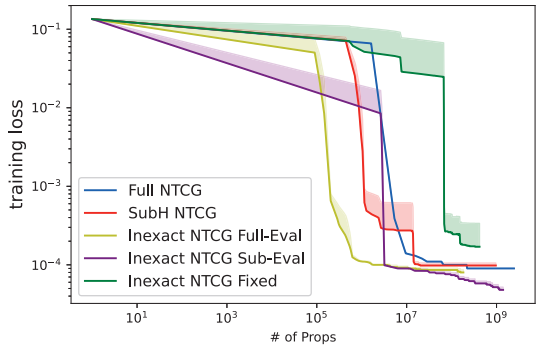

FIG. 3. Comparison between all variants of NTCG on VAE.

3.3 *Variational autoencoder*

We now evaluate the performance of Algorithms 3 and 4 using a more complex setting of variational autoencoder (VAE) model. Our VAE model consists of six fully-connect layers, which are structured as $784 \to 512 \to 256 \to 2 \to 256 \to 512 \to 784$. The intermediate activation and the output truncation functions, are respectively chosen as SoftPlus (Glorot *et al.*, 2011) and Sigmoid (Glorot *et al.*, 2011). We again consider the `MNIST` dataset.

The results are shown in Figure 3. Although we did not fine-tune the fixed step sizes used within `Inexact NTCG Fixed` (as evidenced by its clear nonmonotonic behavior), one can see that `Inexact NTCG Fixed` exhibits competitive performance. Again, as observed previously, `Inexact NTCG Full-Eval` and `Inexact NTCG Sub-Eval` have the fastest convergence rate among all of the variants.

4. Conclusion

We have described extensions of the Newton-CG algorithm proposed by Royer *et al.* (2020) to settings with inexact Hessian and/or gradient information. Algorithm 3 employs approximations to the gradient and Hessian matrix at each step, and this inexact information is used to obtain an approximate Newton direction in Procedure 1. However, to obtain the step size, Algorithm 3 requires exact function values. This issue is partially addressed in Algorithm 4, where fixed step sizes replace the line search. These fixed step sizes are conservative, and they depend on some problem-dependent quantities that are generally unavailable, though known for some important classes of machine learning problems. Still, our approach here can be regarded as a first attempt at incorporating inexact gradient/Hessian evaluations in the Newton-CG algorithm of Royer *et al.* (2020). An improved algorithm would allow for line searches using inexact function evaluations. One might be able to derive such a version using some further assumptions on the inexact function and the inexact gradient, such as those considered in Paquette & Scheinberg (2020), and by introducing randomness into the algorithm and the use of concentration bounds in the analysis. We intend to investigate these topics in future research.

We are especially interested in problems in which the objective has a 'finite-sum' form, so the approximated gradients and Hessians are obtained by sampling randomly from the sum. For all of our proposed variants, we showed that the iteration complexities needed to achieve approximate second-order criticality are essentially the same as that of the exact variants. In particular, a variant that uses a fixed step size, rather than a step chosen adaptively by a backtracking line search, attains the same order of complexity as the other variants, despite never needing to evaluate the function itself.

Our algorithms depend on the randomized Procedure 2 to obtain a negative curvature direction or to certify the approximate second-order optimality. Consequently, unlike the first-order optimality which is guaranteed irrespective of the failure probability in Procedure 2, any point returned by Algorithms 3 and 4 satisfies the approximate second-order optimality condition with a high probability that depends on the failure probability of Procedure 2, and hence can be chosen as desired.

We demonstrate the advantages and shortcomings of the approach, in comparison with other methods, using several test problems.

Funding

F. Roosta was partially supported by the Australian Research Council through a Discovery Early Career Researcher Award (DE180100923) as well as an Industrial Transformation Training Centre for Information Resilience (IC200100022). S. Wright was partially supported by NSF Awards 1740707 and 2023239; DOE ASCR under Subcontract 8F-30039 from Argonne National Laboratory; Award N660011824020 from the DARPA Lagrange Program; and AFOSR under subcontract UTA20-001224 from the University of Texas-Austin. M. Mahoney would also like to acknowledge DARPA, NSF and ONR for providing partial support of this work.

Acknowledgements

We wish to thank two referees for numerous perceptive suggestions that improved this work. One referee suggested an important improvement to Algorithm 3 that we adopted in the revised version of the paper.

References

Beck, A. (2017) *First-Order Methods in Optimization*. MOS-SIAM Series on Optimization. Philadelphia, PA: Society for Industrial and Applied Mathematics.

Bellavia, S., Gurioli, G., Morini, B. & Toint, P. L. (2019) Adaptive regularization algorithms with inexact evaluations for nonconvex optimization. *SIAM J. Optim.*, **29**, 2881–2915.

Bellavia, S. & Gurioli, G. (2022) Stochastic analysis of an adaptive cubic regularization method under inexact gradient evaluations and dynamic Hessian accuracy. *Optimiz. J. Math. Program. Oper. Res.*, **71**, 227–261.

Bertsekas, D. P. (1999) *Nonlinear Programming*. Nashua NH: Athena Scientific.

Blanchet, J., Cartis, C., Menickelly, M. & Scheinberg, K. (2019) Convergence rate analysis of a stochastic trust-region method via supermartingales. *INFORMS J. Optimiz.*, **1**, 92–119.

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

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

Cartis, C., Gould, N. I. M. & Toint, P. L. (2012) Complexity bounds for second-order optimality in unconstrained optimization. *J. Complex.*, **28**, 93–108.

Cartis, C. & Scheinberg, K. (2018) Global convergence rate analysis of unconstrained optimization methods based on probabilistic models. *Math. Program.*, **169**, 337–375.

Chang, C.-C. & Lin, C.-J. (2011) LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Tech.*, **2**, 1–27.

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B. & LeCun, Y. (2015) The loss surfaces of multilayer networks. *Artificial Intelligence and Statistics. PMLR*, **38**, 192–204.

Conn, A. R., Gould, N. I. & Toint, P. L. (2000) *Trust Region Methods*. Philadelphia, PA: SIAM.

Curtis, F. E., Robinson, D. P. & Samadi, M. (2014) A trust region algorithm with a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$ for nonconvex optimization. *COR@ L Technical Report 14T-009*. Bethlehem, PA, USA: Lehigh University.

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

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S. & Bengio, Y. (2014) Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Adv. Neural Inf. Process. Syst.*, **27**, 2933–2941.

Duchi, J., Hazan, E. & Singer, Y. (2011) Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, **12**, 2121–2159.

Ge, R., Huang, F., Jin, C. & Yuan, Y. (2015) Escaping from saddle points-online stochastic gradient for tensor decomposition. *Proceedings of the 28th Conference on Learning Theory,* vol. 40. Paris, France: PMLR, pp. 797–842.

Glorot, X., Bordes, A. & Bengio, Y. (2011) Deep sparse rectifier neural networks. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics,* vol. 15. Fort Lauderdale, FL: PMLR, pp. 315–323.

Gratton, S., Royer, C. W., Vicente, L. N. & Zhang, Z. (2018) Complexity and global rates of trust-region methods based on probabilistic models. *IMA J. Numer. Anal.*, **38**, 1579–1597.

Hillar, C. J. & Lim, L.-H. (2013) Most tensor problems are NP-hard. *J. ACM (JACM)*, **60**, 1–39.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M. & Jordan, M. I. (2017) How to escape saddle points efficiently. *Proceedings of the 34th International Conference on Machine Learning-Volume 70,* vol. 70. Sydney NSW Australia: PMLR, pp. 1724–1732.

Kingma, D. P. & Ba, J. (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Lan, G. (2020) *First-Order and Stochastic Optimization Methods for Machine Learning*. Springer Series in the Data Sciences. Cham, Switzerland: Springer International Publishing.

LeCun, Y. A., Bottou, L., Orr, G. B. & Müller, K.-R. (2012) Efficient backprop. *Neural Networks: Tricks of the Trade*. Berlin, Heidelberg: Springer, pp. 9–48.

Levy, K. Y. (2016) The power of normalization: faster evasion of saddle points. arXiv preprint arXiv:1611.04831.

Lin, Z., Li, H. & Fang, C. (2020) *Accelerated Optimization for Machine Learning: First-Order Algorithms*. Singapore: Springer.

Liu, Y. & Roosta, F. (2021) Convergence of Newton-MR under inexact Hessian information. *SIAM J. Optim.*, **31**, 59–90.

Mishra, S. K. & Giorgi, G. (2008) *Invexity and Optimization,* vol. 88. Berlin, Heidelberg: Springer Science & Business Media.

Murty, K. G. & Kabadi, S. N. (1987) Some NP-complete problems in quadratic and nonlinear programming. *Math. Program.*, **39**, 117–129.

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

Nocedal, J. & Wright, S. J. (2006) *Numerical Optimization*, 2nd edn. New York, NY: Springer Science & Business Media.

Paquette, C. & Scheinberg, K. (2020) A stochastic line search method with expected complexity analysis. *SIAM J. Optim.*, **30**, 349–376.

Roosta, F., Liu, Y., Xu, P. & Mahoney, M. W. (2018) Newton-MR: Newton's method without smoothness or convexity. arXiv preprint arXiv:1810.00303.

Roosta, F. & Mahoney, M. W. (2019) Sub-sampled Newton methods. *Math. Program.*, **174**, 293–326.

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

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

Saxe, A. M., McClelland, J. L. & Ganguli, S. (2013) Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. arXiv preprint arXiv:1312.6120.

Shewchuk, J. R. (1994) An introduction to the conjugate gradient method without the agonizing pain. *Technical Report. Carnegie Mellon University, USA*.

Steihaug, T. (1983) The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, **20**, 626–637.

Tripuraneni, N., Stern, M., Jin, C., Regier, J. & Jordan, M. I. (2018) Stochastic cubic regularization for fast nonconvex optimization. *Advances in Neural Information Processing Systems*. Montreal, Canada: PMLR, pp. 2899–2908.

Wright, S. J. & Recht, B. (2021) *Optimization for Data Analysis*. Cambridge, UK: Cambridge University Press (In press).

Xie, Y. & Wright, S. J. (2021) Complexity of projected Newton methods for bound-constrained optimization. arXiv preprint arXiv:2103.15989.

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

Xu, P., Roosta, F. & Mahoney, M. W. (2020b) Second-order optimization for non-convex machine learning: An empirical study. *Proceedings of the 2020 SIAM International Conference on Data Mining*. Cincinnati, Ohio: SIAM, pp. 199–207.

Yao, Z., Xu, P., Roosta, F. & Mahoney, M. W. (2020) Inexact non-convex Newton-type methods. *INFORMS J. Optimiz*. https://doi.org/10.1287/ijoo.2019.0043.

Zhang, R., Mei, Y., Shi, J. & Xu, H. (2019) Robustness and tractability for non-convex m-estimators. arXiv preprint arXiv:1906.02272.