SIAM J. OPTIM.
Vol. 33, No. 3, pp. 2191–2221

WORST-CASE COMPLEXITY OF TRACE WITH INEXACT SUBPROBLEM SOLUTIONS FOR NONCONVEX SMOOTH OPTIMIZATION*

FRANK E. CURTIS† AND QI WANG†

Abstract. An algorithm for solving nonconvex smooth optimization problems is proposed, analyzed, and tested. The algorithm is an extension of the trust-region algorithm with contractions and expansions (TRACE) [F. E. Curtis, D. P. Robinson, and M. Samadi, *Math. Program.*, 162 (2017), pp. 1–32]. In particular, the extension allows the algorithm to use inexact solutions of the arising subproblems, which is an important feature for solving large-scale problems. Inexactness is allowed in a manner such that the optimal iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$ for attaining an ϵ-approximate first-order stationary point is maintained, while the worst-case complexity in terms of Hessian-vector products may be significantly improved as compared to the original TRACE. Numerical experiments show the benefits of allowing inexact subproblem solutions and that the algorithm compares favorably to state-of-the-art techniques.

Key words. nonlinear optimization, nonconvex optimization, worst-case iteration complexity, worst-case evaluation complexity, trust-region methods

MSC codes. 49M37, 65K05, 65K10, 65Y20, 68Q25, 90C30, 90C60

DOI. 10.1137/22M1492428

1. Introduction.

There are a variety of algorithmic methodologies for solving nonconvex smooth optimization problems that offer state-of-the-art performance when solving broad subclasses of such problems. Among these, a few offer a worst-case performance guarantee for achieving approximate first-order stationarity that is optimal with respect to a class of second-order-derivative-based methods for minimizing sufficiently smooth objective functions [7]. These include certain cubic-regularization, quadratic-regularization, line-search, and trust-region methods; see section 1.2.

In this paper, we propose, analyze, and provide the results of numerical experiments with an extended version of the *trust-region algorithm with contractions and expansions* (TRACE) from [14], which was the first trust-region method to attain the aforementioned optimal iteration complexity guarantees. In particular, the algorithm that we propose overcomes the main deficiency of TRACE, namely, that TRACE requires exact solutions of the arising trust-region subproblems, which is impractical in large-scale settings. Our algorithm overcomes this deficiency by employing an iterative linear algebra technique—namely, a Krylov subspace method inspired by [20]—for solving the arising subproblems and allowing the "outer" algorithm for solving the original problem to use inexact solutions from the "inner" algorithm for solving the trust-region subproblems. This represents a response to the following conjecture from [14]: "We expect that such variations of our algorithm can be designed that maintain our global convergence guarantees... our worst-case complexity bounds and local convergence guarantees." In fact, our proposed algorithm goes beyond this conjecture. We not only show that our approach maintains the convergence and worst-case

*Received by the editors April 25, 2022; accepted for publication (in revised form) March 23, 2023; published electronically August 18, 2023.
https://doi.org/10.1137/22M1492428
Funding: The work of the authors was supported by the U.S. National Science Foundation under awards CCF-2008484 and CCF-2139735.
†Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015 USA (frank.e.curtis@gmail.com, qiw420@lehigh.edu).

complexity guarantees of TRACE but also show that our proposed enhancement of TRACE achieves strong worst-case complexity properties in terms of the overall required number of Hessian-vector products, which are the most expensive operations required when solving many large-scale problems. We also show that our proposed enhancement achieves the same local convergence rate as TRACE if appropriate restrictions, stated explicitly in our analysis, are imposed on the inexactness.

Our numerical experiments show that our algorithm offers practical benefits over other optimal-worst-case-complexity methods for solving nonconvex smooth optimization problems. We also demonstrate that our approach offers the computational flexibility in terms of the trade-offs between derivative evaluations and Hessian-vector products that should be expected of any such method that allows inexact subproblem solutions. In particular, with more exact subproblem solutions, our algorithm often requires fewer derivative evaluations at the expense of more Hessian-vector products, whereas with more inexact subproblem solutions, it often requires fewer Hessian-vector products at the expense of more derivative evaluations. This allows any user of our algorithm to tailor its use depending on the relative costs of these operations.

1.1. Notation, problem formulation, and assumptions. We use $\mathbb{R}$ to denote the set of real numbers, $\mathbb{R}_{\geq 0}$ (resp., $\mathbb{R}_{>0}$) to denote the set of nonnegative (resp., positive) real numbers, $\mathbb{R}^n$ to denote the set of n-dimensional real vectors, $\mathbb{R}^{m\times n}$ to denote the set of m-by-n-dimensional real matrices, $\mathbb{S}^n \subset \mathbb{R}^{n\times n}$ to denote the set of n-by-n-dimensional real symmetric matrices, and $\mathbb{N}$ to denote the set of nonnegative integers. We use I to denote the identity matrix and use e_j for $j \in \mathbb{N}\backslash\{0\}$ to denote the jth column of the identity matrix, where in each case the dimension of the object is determined by the context in which it appears. We use the function $|\cdot|$ to take the absolute value of a real number and use the function $\|\cdot\|$ to take the 2-norm of a real vector or to take the induced 2-norm of a real matrix. Given real numbers a and b, we use $a \perp b$ to mean that $ab = 0$. Given $H \in \mathbb{S}^n$, we use $H \succ 0$ (resp., $H \succeq 0$) to indicate that H is positive definite (resp., semidefinite).

Given functions $\phi : \mathbb{R} \to \mathbb{R}_{\geq 0}$ and $\varphi : \mathbb{R} \to \mathbb{R}_{\geq 0}$, the expression $\phi(\cdot) = \mathcal{O}(\varphi(\cdot))$ means that there exists $c \in \mathbb{R}_{>0}$ such that $\phi(\cdot) \leq c\varphi(\cdot)$. Similarly, given positive real number sequences $\{\phi_k\}$ and $\{\varphi_k\}$, the expression $\phi_k = \mathcal{O}(\varphi_k)$ means that there exists $c \in \mathbb{R}_{>0}$ such that $\phi_k \leq c\varphi_k$ for all $k \in \mathbb{N}$. If, in addition, the sequences have the property that $\{\phi_k/\varphi_k\} \to 0$, then one writes $\phi_k = o(\varphi_k)$.

Our problem of interest is the minimization problem

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

where $f : \mathbb{R}^n \to \mathbb{R}$ satisfies Assumption 1.1, stated below. The gradient and Hessian functions for f are denoted by $g := \nabla f : \mathbb{R}^n \to \mathbb{R}$ and $H := \nabla^2 f : \mathbb{R}^n \to \mathbb{S}^n$, respectively. Given the kth iterate in an algorithm for solving (1.1)—call it $x_k \in \mathbb{R}^n$— we define $f_k := f(x_k)$, $g_k := g(x_k) \equiv \nabla f(x_k)$, and $H_k := H(x_k) \equiv \nabla^2 f(x_k)$. We also apply a subscript to refer to other quantities corresponding to the kth iteration; e.g., the iterate displacement (i.e., step) is denoted as $s_k \in \mathbb{R}^n$. Our algorithm involves subroutines that have their own iteration indices, and we use additional subscripts to keep track of quantities associated with the inner iterations of these subroutines.

Assumption 1.1 is made throughout the paper. For TRACE in [14], a guarantee of convergence from remote starting points is first proved under a weaker assumption—namely, without the assumption of Lipschitz continuity of the Hessian function—prior to worst-case iteration complexity guarantees being proved under an assumption on par with Assumption 1.1. We claim that the same could be done for the algorithm proposed in this paper, but for the sake of brevity we jump immediately to

Assumption 1.1 in order to prove worst-case complexity properties. As stated later, these properties, in turn, ensure convergence from remote starting points.

Assumption 1.1 refers, for each index tuple $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ generated by our algorithm, to the iterate $x_k \in \mathbb{R}^n$ and trial step $Q_{k,j}t_{k,j,l} \in \mathbb{R}^n$. The fact that the assumption refers to these algorithmic quantities should not be seen as a deficiency of our analysis. After all, our algorithm guarantees monotonic nonincrease of the objective values, meaning that $\{x_k\}$ is contained in the sublevel set for f with respect to the initial value $f(x_0)$, i.e., $\mathcal{L}_{f,0} := \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$. In addition, with each accepted step, the algorithm requires a decrease in the objective that is proportional to the cubed norm of the step, so with an objective that is bounded below, it is reasonable to assume that the accepted steps are bounded in norm, which, in turn, means that it is reasonable to assume (by construction of our algorithm) that all trial steps are bounded in norm. Assuming that is the case, the open convex set mentioned in the assumption would itself be contained in the Minkowski sum of $\mathcal{L}_{f,0}$ and a bounded set, in light of which the assumption is standard for smooth optimization.

Assumption 1.1. The function $f : \mathbb{R}^n \to \mathbb{R}$ is twice-continuously differentiable and bounded below by a real number $f_{\inf} \in \mathbb{R}$ over $\mathbb{R}^n$. In addition, both the gradient function $g : \mathbb{R}^n \to \mathbb{R}^n$ and Hessian function $H : \mathbb{R}^n \to \mathbb{S}^n$ are Lipschitz continuous with Lipschitz constants denoted by $g_{\text{Lip}} \in \mathbb{R}_{>0}$ and $H_{\text{Lip}} \in \mathbb{R}_{>0}$, respectively, in an open convex set containing x_k and $x_k + Q_{k,j}t_{k,j,l}$ for all generated $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$.

A consequence of the Lipschitz continuity of the gradient in Assumption 1.1 is that the Hessian matrix H_k is bounded in norm for all $k \in \mathbb{N}$ in the sense that there exists $H_{\max} \in \mathbb{R}_{>0}$ such that $\|H_k\| \leq H_{\max}$ for all $k \in \mathbb{N}$.

1.2. Literature review. Our focus in this paper is on worst-case complexity bounds for a second-order-derivative-based algorithm to reach an iterate x_k that is ϵ-approximate first-order stationary (we call this property ϵ-stationary throughout the paper) with respect to (1.1) in the sense that

$$\|\nabla f(x_k)\| \leq \epsilon. \tag{1.2}$$

Some research articles have also considered worst-case complexity bounds for achieving second- or even higher-order stationarity, but since our focus in this paper is on large-scale settings in which requiring such guarantees is impractical, we consider such complexity bounds outside the scope of our study.

Trust-region methods that employ a traditional updating scheme for the trust-region radius based on actual-to-quadratic-model-predicted-reduction ratios (see, e.g., [10, 31]) are known to have a worst-case *iteration* complexity (and, correspondingly, *function-* and *derivative-evaluation* complexities) of $\mathcal{O}(\epsilon^{-2})$ for achieving ϵ-stationarity [6, 12, 24]. Importantly, this complexity is known to be tight for both first- and second-order variants of such methods [6]. It was first shown by Nesterov and Polyak that cubic-regularization of a second-order method—an idea that has appeared as far back as Griewank [25]—can achieve an improved iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$; see [30]. This complexity for achieving ϵ-stationarity is now known to be optimal with respect to a class of second-order-derivative-based methods for minimizing sufficiently smooth objectives [6]. After [30], practical variants of the idea subsequently appeared in papers by Cartis, Gould, and Toint [8, 9] in the form of the adaptive-regularization-using-cubics (ARC) method. Since that time, a few other schemes have been developed that build upon cubic- or even quadratic-regularization techniques; see, e.g., [3, 15, 18, 19, 22]. These ideas have also been extended to high-order regularization of higher-order methods in order to achieve improved complexity

bounds; see, e.g., the work by Birgin et al. in [2], where it is shown that a pth-order method with $(p+1)$th-order regularization can achieve an iteration complexity of $\mathcal{O}(\epsilon^{-(p+1)/p})$.

Our work in this paper is motivated by the fact that trust-region methods have proved to be effective algorithms in practice for solving (large-scale) nonconvex smooth optimization problems. Indeed, certain trust-region methods that have been designed without worst-case complexity properties are often found to be some of the most reliable and efficient methods in practical situations. Therefore, we contend that it is of interest to explore trust-region methods that do not deviate too much from contemporary schemes yet offer optimal worst-case complexity properties. The first such trust-region method to achieve $\mathcal{O}(\epsilon^{-3/2})$ iteration complexity with respect to ϵ-stationarity was TRACE [14]. Another approach, which tries to adhere closely to the popular combination of a trust-region method that employs the linear conjugate gradient (CG) method to solve the arising subproblems, is that in [13]; see also the prior line-search method proposed by Royer and Wright in [33].

As previously mentioned, our work in this paper is motivated by the goal to improve the *computational* complexity of TRACE in large-scale settings where matrix factorizations and/or Hessian-vector products can dominate the computational expense. The use of iterative linear algebra techniques to exploit potentially inexact subproblem solutions has been a topic of research for decades. Traditional line-search and trust-region methods that use CG to solve the arising subproblems (approximately) have been studied and implemented widely [34, 35]. It is well known that with sufficiently exact subproblem solutions, such algorithms can attain the superlinear or quadratic rates of local convergence of Newton's method [16]. Similar guarantees have also been shown for ARC [8, 9]—see also [2] for the use of inexact subproblem solutions in higher-order regularization schemes—although the analysis in [8, 9] does not delve into the computational (i.e., Hessian-vector-product) complexity of the proposed scheme that allows inexact subproblem solutions. Particularly in the case of trust-region methods when one aims to solve the subproblems to arbitrary accuracy, the use of the Lanczos method has been well studied [20], for which it is known that if the solution of a trust-region subproblem in n variables lies on the boundary of the trust-region radius, then the subproblem is equivalent to an extremal eigenvalue problem of a matrix of size $2n$ [1]. Convergence of the Lanczos method for estimating eigenvalues has been analyzed in [27, 28], the results of which have been used in the analysis of various optimization algorithms; see, e.g., [5, 13, 32, 33]. Complexity guarantees for the Lanczos method specifically for trust-region methods have been studied in [4, 23, 26, 36]. Among these works, an upper bound on the (subproblem) objective function error at every Lanczos iteration is proved in [4], and a bound on the objective difference between any two iterations is derived in [23]. An upper bound on the residual at every iteration is proved in [23]; this result is used directly in this paper to derive a complexity bound in terms of Hessian-vector products of our algorithm. Finally, we remark that the algorithm proposed in [15] can be viewed as one possible extension of TRACE that employs inexact subproblem solutions. In fact, it is more appropriately referred to as a generalization of TRACE since other algorithms, such as ARC, can be viewed as special cases of it. However, as in [8, 9], the analysis in [15] does not prove computational complexity guarantees for the generalized framework, which is a primary motivation of the work in this paper.

Along with our concluding remarks in section 5, we provide some additional discussion about the worst-case complexity properties of our algorithm and how they compare with those of other algorithms in the literature.

1.3. Contributions. The work in this paper builds on the ideas and analyses provided in the aforementioned literature but offers a unique contribution since we provide the first inexact variant of TRACE, a method that we call I-TRACE, that offers iteration and gradient evaluation complexity bounds that match those of TRACE. We also use results about the complexity of the Lanczos algorithm to show that I-TRACE offers state-of-the-art complexity in terms of Hessian-vector products when solving large-scale problem instances. Our theoretical analyses are backed by empirical evidence showing that our proposed I-TRACE method offers computational flexibility beyond that offered by TRACE and compares favorably against implementations of two other optimal-worst-case-complexity methods that also allow inexact subproblem solutions. We attribute this behavior to the fact that I-TRACE adheres closely to a traditional trust-region strategy in that to achieve optimal iteration complexity, it adaptively uses a combination of explicit and implicit regularizations of the Hessian matrices in the arising subproblems.

1.4. Organization. Section 2 contains a description of our algorithm and its associated subroutines. (Our description includes well-known characterizations and properties of Krylov subspace, specifically Lanczos-based, iterative methods for solving subproblems arising in optimization algorithms, for which we refer the reader to [11, 20, 36]). Section 3 contains our convergence and worst-case complexity analyses of the algorithm. The results of numerical experiments are provided in section 4, and concluding remarks are provided in section 5.

2. Algorithm description. Each iteration of I-TRACE involves the minimization of a second-order Taylor series model of f at the current iterate within a trust region; specifically, in iteration $k \in \mathbb{N}$, the model $m_k : \mathbb{R}^n \to \mathbb{R}$ is defined by

$$m_k(s) = f_k + g_k^T s + \tfrac{1}{2} s^T H_k s.$$

Building on TRACE, the trust region is defined either explicitly through a trust-region radius $\delta \in \mathbb{R}_{>0}$ and a trust-region constraint of the form $\|s\| \leq \delta$ or implicitly through a regularization parameter $\lambda \in \mathbb{R}_{>0}$ and a regularization term $\frac{1}{2}\lambda\|s\|^2$, where λ is sufficiently large such that $H_k + \lambda I \succ 0$, which, in turn, means that the regularized model $m_k(\cdot) + \frac{1}{2}\lambda\|\cdot\|^2 = f_k + g_k^T(\cdot) + \frac{1}{2}(\cdot)^T(H_k + \lambda I)(\cdot)$ is strongly convex.

However, unlike TRACE, the main idea behind I-TRACE is to allow an approximate subproblem solution to be considered acceptable. Specifically, in each iteration $k \in \mathbb{N}$, the algorithm might only consider, for some $j \in \{0, \ldots, n-1\}$, the solution of a subproblem over the jth-order Krylov subspace defined by g_k and H_k, namely,

$$\mathcal{K}_{k,j} := \operatorname{span}\{g_k, H_k g_k, \ldots, H_k^j g_k\}.$$

Using the Lanczos process, I-TRACE iteratively constructs orthonormal bases for such subspaces for increasing j as needed. Let such a basis be given by

$$Q_{k,j} := \begin{bmatrix} q_{k,0} & q_{k,1} & \cdots & q_{k,j} \end{bmatrix} \in \mathbb{R}^{n \times (j+1)}.$$

With this basis constructed using Lanczos, one finds for any $(k, j) \in \mathbb{N} \times \{0, \ldots, n-1\}$ that there exists tridiagonal $T_{k,j} \in \mathbb{R}^{(j+1)\times(j+1)}$ such that with $\gamma_{k,0} \leftarrow \|g_k\|$ one has

$$T_{k,j} = Q_{k,j}^T H_k Q_{k,j} \quad \text{and} \quad \gamma_{k,0} e_1 = Q_{k,j}^T g_k.$$

Overall, for generated $(k,j) \in \mathbb{N} \times \{0,\ldots,n-1\}$, the algorithm considers the trust-region subproblem for a given trust-region radius $\delta \in \mathbb{R}_{>0}$ defined as

$$\mathcal{S}_{k,j}(\delta) : \left[\min_{t \in \mathbb{R}^{j+1}} \ \gamma_{k,0} e_1^T t + \frac{1}{2} t^T T_{k,j} t \text{ s.t. } \|t\| \leq \delta \right]$$

and/or the regularized subproblem for a given regularization parameter $\lambda \in \mathbb{R}_{>0}$ (sufficiently large such that $T_{k,j} + \lambda I \succ 0$) defined as

$$\mathcal{R}_{k,j}(\lambda) : \left[\min_{t \in \mathbb{R}^{j+1}} \ \gamma_{k,0} e_1^T t + \tfrac{1}{2} t^T (T_{k,j} + \lambda I) t \right].$$

(We drop the constant objective term f_k in both subproblems since it does not affect the solution sets.) The key feature of these subproblems is the fact that the matrix $T_{k,j}$ is tridiagonal, meaning that both can be solved to high accuracy (i.e., exactly for the purposes of our theoretical analysis) in an efficient manner. In particular, $\mathcal{S}_{k,j}(\delta)$ can be solved using the Moré–Sorensen method [29], while $\mathcal{R}_{k,j}(\lambda)$ can be solved by solving the (nonsingular) tridiagonal system $(T_{k,j} + \lambda I)t = -\gamma_{k,0} e_1$. For future reference, we note that necessary and sufficient conditions for global optimality with respect to $\mathcal{S}_{k,j}(\delta)$ are that $(t_{k,j}, \lambda_{k,j}) \in \mathbb{R}^{j+1} \times \mathbb{R}$ is globally optimal if and only if

$$\gamma_{k,0} e_1 + (T_{k,j} + \lambda_{k,j} I) t_{k,j} = 0, \tag{2.1a}$$

$$(T_{k,j} + \lambda_{k,j} I) \succeq 0, \tag{2.1b}$$

$$\text{and} \quad 0 \leq \lambda_{k,j} \perp (\delta - \|t_{k,j}\|) \geq 0. \tag{2.1c}$$

Each iteration $k \in \mathbb{N}$ of I-TRACE begins by computing a solution of $\mathcal{S}_{k,j}(\delta_k)$ for some $\delta_k \in \mathbb{R}_{>0}$ and sufficiently large $j \in \mathbb{N}$. In particular, I-TRACE employs the *truncated Lanczos trust-region* (TLTR) algorithm (see [11, Algorithm 5.2.1]) that solves trust-region subproblems over Krylov subspaces of increasing size (i.e., increasing j) until a termination condition holds. We state our variant of the algorithm in detail as Algorithm 2.1, which generates the aforementioned quantities $\{Q_{k,j}\}_{j \in \mathbb{N}}$ and $\{T_{k,j}\}_{j \in \mathbb{N}}$ as well as some auxiliary values required for Lanczos. For our purposes with I-TRACE, the termination conditions that we use are written in the full space as

$$g_k + (H_k + \lambda_{k,j} I) s_{k,j} = r_{k,j}, \tag{2.2a}$$

$$r_{k,j}^T s_{k,j} \leq 0, \tag{2.2b}$$

$$s_{k,j}^T (H_k + \lambda_{k,j} I) s_{k,j} \geq 0, \tag{2.2c}$$

$$\text{and} \quad 0 \leq \lambda_{k,j} \perp (\delta_k - \|s_{k,j}\|) \geq 0 \tag{2.2d}$$

along with

$$\text{either} \qquad \|r_{k,j}\| \leq \xi_1 \|s_{k,j}\|^2 \tag{2.3a}$$

$$\text{or both} \qquad \|r_{k,j}\| \leq \xi_2 \min\{1, \|s_{k,j}\|\} \|g_k\| \tag{2.3b}$$

$$\text{and} \qquad 1 \leq \xi_3 \min\{1, \|s_{k,j}\|\} \|T_{k,j} + \lambda_{k,j} I\|, \tag{2.3c}$$

where $(\xi_1, \xi_2, \xi_3) \in \mathbb{R}_{>0} \times (0,1) \times \mathbb{R}_{>0}$ are user-prescribed parameters. (For simplicity of software implementation, I-TRACE can always require (2.3a) and simply ignore (2.3b)–(2.3c), but for the sake of generality in our analysis and computational flexibility, we show that imposing (2.3) leads to the same complexity guarantees. This is of

interest since, in some situations in practice, the conditions in (2.3b)–(2.3c) may be less restrictive than (2.3a).) The algorithm may also impose tighter residual conditions that are similar to ones imposed in inexact Newton methods [16] to achieve a fast rate of local convergence; these are specified along with our analysis in section 3.3.

Generally speaking, smaller values of ξ_1 and ξ_2 imply that more accurate subproblem solutions must be computed, which, in turn, may mean that fewer iterations are required to satisfy (1.2) for a given ϵ. However, this likely comes at the cost of more computational effort in each iteration. On the other hand, a larger value of ξ_1 and a value of ξ_2 near 1 imply that less accurate subproblem solutions must be computed. This may reduce the per-iteration computational cost, although it may also mean that more iterations need to be performed to satisfy (1.2). As for other inexact algorithms, this presents a trade-off that we explore in our numerical experiments in section 4, specifically with the values $(\xi_1, \xi_2) \in \{(0.1, 0.01), (1, 0.1), (9, 0.9)\}$, which represent a wide range of possible choices in practice. As for ξ_3 and condition (2.3c), these are needed for theoretical reasons (see Lemma 3.18 and the discussion after that lemma to understand the role that (2.3c) plays), but we suspect that it may be beneficial in practice to set ξ_3 to a large value so that (2.3c) is satisfied easily.

Algorithm 2.1 generates, for each generated value of $j \in \{0, \ldots, n-1\}$, the primal-dual solution $(t_{k,j}, \lambda_{k,j})$ of $\mathcal{S}_{k,j}(\delta_k)$, the primal solution of which defines the full-space trial step $s_{k,j} \in \mathbb{R}^n$, which, in turn, defines the residual vector $r_{k,j} \in \mathbb{R}^n$ by (2.2a). If $r_{k,j} = 0$, then (2.2) is essentially the set of necessary and sufficient conditions for the global minimization of $m_k(s)$ over $s \in \mathbb{R}^n$ such that $\|s\| \leq \delta_k$, with the only difference being the relaxed condition that $s_{k,j}^T(H_k + \lambda_{k,j} I)s_{k,j} \geq 0$ rather than $(H_k + \lambda_{k,j}) \succeq 0$. We show in our theoretical analysis that, by the construction of Algorithm 2.1, the conditions in (2.2) are satisfied for all generated values of j, meaning that the only conditions that need to be checked explicitly are those in (2.3), the first of which is guaranteed to hold at iteration $n-1$ if it is not satisfied earlier. Our analysis shows that, in the reduced space, (2.3) is equivalent (since $\mu_{k,j} = \|r_{k,j}\|$, $\|t_{k,j}\| = \|s_{k,j}\|$, and $\gamma_{k,0} = \|g_k\|$) to

$$\begin{aligned}
&\text{(2.4a)} && \text{either} && \mu_{k,j} \leq \xi_1 \|t_{k,j}\|^2 \\
&\text{(2.4b)} && \text{or both} && \mu_{k,j} \leq \xi_2 \min\{1, \|t_{k,j}\|\}\gamma_{k,0} \\
&\text{(2.4c)} && \text{and} && 1 \leq \xi_3 \min\{1, \|t_{k,j}\|\}\|T_{k,j} + \lambda_{k,j} I\|,
\end{aligned}$$

which are the conditions that are actually checked in the algorithm. (As previously mentioned for (2.3a), for simplicity, I-TRACE can always require (2.4a) and simply ignore (2.4b)–(2.4c), but for the sake of generality in our analysis and computational flexibility, we show that imposing (2.4) as it is stated leads to the same complexity guarantees.)

Upon completion of the call to Algorithm 2.1 in iteration $k \in \mathbb{N}$, I-TRACE turns to determine whether the current trial solution should be accepted, whether an alternative trial solution should be computed *over the current Krylov subspace* after an expansion and/or contraction(s) of the trust-region radius, or whether the Krylov subspace should be increased in dimension. This is done by first calling a subroutine that we call "find decrease step" (FDS), stated as Algorithm 2.2. This method checks conditions derived from those in TRACE and ultimately produces a step—potentially after expansion and/or contraction(s) of the trust-region radius—that offers sufficient decrease in the objective and a ratio between the subproblem's dual solution and the norm of the subproblem's primal solution that is sufficiently small, *all while keeping*

Algorithm 2.1 TLTR (an adaptation of [11, Algorithm 5.2.1]).

Parameters: $(\xi_1, \xi_2, \xi_3) \in \mathbb{R}_{>0} \times (0,1) \times \mathbb{R}_{>0}$ from I-TRACE (Algorithm 2.3)
Input: $y_{k,0}$ $(= g_k)$, $\gamma_{k,0}$ $(= \|g_k\|)$, H_k, δ_k
1: $q_{k,-1} \leftarrow 0$, $Q_{k,-1} \leftarrow [\,]$
2: **for** $j = 0, 1, \ldots$ **do**
3: $\quad q_{k,j} \leftarrow \frac{1}{\gamma_{k,j}} y_{k,j}$
4: $\quad Q_{k,j} \leftarrow \begin{bmatrix} Q_{k,j-1} & q_{k,j} \end{bmatrix}$
5: $\quad \theta_{k,j} \leftarrow q_{k,j}^T H_k q_{k,j}$
6: $\quad$ **if** $j = 0$ **then**
7: $\quad\quad T_{k,j} \leftarrow \begin{bmatrix} \theta_{k,0} \end{bmatrix}$
8: $\quad$ **else**
9: $\quad\quad T_{k,j} \leftarrow \begin{bmatrix} & & & 0 \\ & T_{k,j-1} & & \vdots \\ & & & \gamma_{k,j} \\ 0 & \cdots & \gamma_{k,j} & \theta_{k,j} \end{bmatrix}$
10: $\quad$ **end if**
11: $\quad y_{k,j+1} \leftarrow H_k q_{k,j} - \theta_{k,j} q_{k,j} - \gamma_{k,j} q_{k,j-1}$
12: $\quad \gamma_{k,j+1} \leftarrow \|y_{k,j+1}\|$
13: $\quad$ compute $(t_{k,j}, \lambda_{k,j})$ by solving $\mathcal{S}_{k,j}(\delta_k)$ (which gives $s_{k,j} = Q_{k,j} t_{k,j}$)
14: $\quad \mu_{k,j} \leftarrow \gamma_{k,j+1} |e_{j+1}^T t_{k,j}|$ $(= \|r_{k,j}\|$ where $r_{k,j} = g_k + (H_k + \lambda_{k,j} I) s_{k,j})$
15: $\quad$ **if** (2.4) (meaning (2.3)) holds **then** since (2.2) also holds (see Lemma 3.2)
16: $\quad\quad$ **break**
17: $\quad$ **end if**
18: **end for**
19: **return** $(j, t_{k,j}, \lambda_{k,j}, Q_{k,j}, T_{k,j}, y_{k,j+1}, \gamma_{k,j+1}, \mu_{k,j})$

the Krylov subspace fixed. The notion of sufficient decrease in the objective, as in TRACE, uses a function-decrease-to-step-norm ratio of the form

$$\rho_k(s) := \frac{f_k - f(x_k + s)}{\|s\|^3}. \tag{2.5}$$

See [14, section 2.4] for motivation for the use of this ratio for this purpose; in short, using a step acceptance ratio of this form ensures that accepted steps yield a reduction in the objective on the order that is needed to achieve optimal complexity.

As previously mentioned, the matrix $T_{k,j}$ in any call to FDS is tridiagonal, meaning that the arising subproblems (i.e., instances of $\mathcal{S}_{k,j}$ and/or $\mathcal{R}_{k,j}$) can be solved accurately in an efficient manner. The only aspect of the algorithm that might raise suspicion is the computation requested in lines 15–16. However, as for TRACE (see [14, Appendix]), this computation is always well posed (see Lemma 3.3 in the next subsection) and is *no more expensive* than solving an instance of $\mathcal{S}_{k,j}$ or $\mathcal{R}_{k,j}$.

If the full-space solution corresponding to the output from FDS maintains the desired level of accuracy in the full space (recall (2.4)), then—since it has already been shown to yield sufficient decrease and a sufficiently small ratio between the subproblem's dual solution and the norm of the step—I-TRACE accepts the step and proceeds to the next iteration. Otherwise, the dimension of the Krylov subspace is increased—again using the Lanczos process—and FDS is called again to produce a new trial step. These details can be seen in I-TRACE; see Algorithm 2.3.

Algorithm 2.2 Find decrease step (FDS).

Parameters: $\eta \in (0,1)$, $\underline{\sigma} \in (0,\infty)$, $\overline{\sigma} \in (\underline{\sigma},\infty)$, $\gamma_C \in (0,1)$, and $\gamma_\lambda \in (1,\infty)$ from I-TRACE (Algorithm 2.3)

Input: $t_{k,j,0}$, $\lambda_{k,j,0}$, $Q_{k,j}$, $T_{k,j}$, $\gamma_{k,0}$, $\gamma_{k,j+1}$, $\mu_{k,j,0}$, $\delta_{k,j,0}$, $\sigma_{k,j,0}$

for $l = 0, 1, \ldots$ **do**
  if $\rho_k(Q_{k,j}t_{k,j,l}) \geq \eta$ and $\frac{\lambda_{k,j,l}}{\|t_{k,j,l}\|} \leq \sigma_{k,j,l}$ **then** [decrease step found]
    return $(t_{k,j,l}, \lambda_{k,j,l}, \mu_{k,j,l}, \delta_{k,j,t}, \sigma_{k,j,l})$
  else if $\rho_k(Q_{k,j}t_{k,j,l}) \geq \eta$ and $\frac{\lambda_{k,j,l}}{\|t_{k,j,l}\|} > \sigma_{k,j,l}$ **then** [expand trust region]
    $\delta_{k,j,l+1} \leftarrow \frac{\lambda_{k,j,l}}{\sigma_{k,j,l}}$
    compute $(t_{k,j,l+1}, \lambda_{k,j,l+1})$ by solving $\mathcal{S}_{k,j}(\delta_{k,j,l+1})$
    $\sigma_{k,j,l+1} \leftarrow \sigma_{k,j,l}$
  else (i.e., $\rho_k(Q_{k,j}t_{k,j,l}) < \eta$) [contract trust region]
    if $\lambda_{k,j,l} < \underline{\sigma}\|t_{k,j,l}\|$ **then**
      $\hat{\lambda}_{k,j,l+1} \leftarrow \lambda_{k,j,l} + (\underline{\sigma}\gamma_{k,0})^{1/2}$
      compute $\hat{t}_{k,j,l+1}$ by solving $\mathcal{R}_{k,j}(\hat{\lambda}_{k,j,l+1})$
      if $\frac{\hat{\lambda}_{k,j,l+1}}{\|\hat{t}_{k,j,l+1}\|} \leq \overline{\sigma}$ **then**
        $(t_{k,j,l+1}, \lambda_{k,j,l+1}, \delta_{k,j,l+1}) \leftarrow (\hat{t}_{k,j,l+1}, \hat{\lambda}_{k,j,l+1}, \|\hat{t}_{k,j,l+1}\|)$
      else (i.e., $\frac{\hat{\lambda}_{k,j,l+1}}{\|\hat{t}_{k,j,l+1}\|} > \overline{\sigma}$)
        compute $\bar{\lambda}_{k,j,l+1} \in (\lambda_{k,j,l}, \hat{\lambda}_{k,j,l+1})$ so that the solution
          $\bar{t}_{k,j,l+1}$ of $\mathcal{R}_{k,j}(\bar{\lambda}_{k,j,l+1})$ yields $\underline{\sigma} < \frac{\bar{\lambda}_{k,j,l+1}}{\|\bar{t}_{k,j,l+1}\|} < \overline{\sigma}$
        $(t_{k,j,l+1}, \lambda_{k,j,l+1}, \delta_{k,j,l+1}) \leftarrow (\bar{t}_{k,j,l+1}, \bar{\lambda}_{k,j,l+1}, \|\bar{t}_{k,j,l+1}\|)$
      end if
    else (i.e., $\lambda_{k,j,l} \geq \underline{\sigma}\|t_{k,j,l}\|$)
      $\hat{\lambda}_{k,j,l+1} \leftarrow \gamma_\lambda \lambda_{k,j,l}$
      compute $\hat{t}_{k,j,l+1}$ by solving $\mathcal{R}_{k,j}(\hat{\lambda}_{k,j,l+1})$
      if $\|\hat{t}_{k,j,l+1}\| \geq \gamma_C \delta_{k,j,l}$ **then**
        $(t_{k,j,l+1}, \lambda_{k,j,l+1}, \delta_{k,j,l+1}) \leftarrow (\hat{t}_{k,j,l+1}, \hat{\lambda}_{k,j,l+1}, \|\hat{t}_{k,j,l+1}\|)$
      else (i.e., $\|\hat{t}_{k,j,l+1}\| < \gamma_C \delta_{k,j,l}$)
        $\delta_{k,j,l+1} \leftarrow \gamma_C \delta_{k,j,l}$
        compute $(t_{k,j,l+1}, \lambda_{k,j,l+1})$ by solving $\mathcal{S}_{k,j}(\delta_{k,j,l+1})$
      end if
    end if
    $\sigma_{k,j,l+1} \leftarrow \max\left\{\sigma_{k,j,l}, \frac{\lambda_{k,j,l+1}}{\|t_{k,j,l+1}\|}\right\}$
  end if
  $\mu_{k,j,l+1} \leftarrow \gamma_{k,j+1}|e_{j+1}^T t_{k,j,l+1}|$
end for

We close this section by noting that TRACE also generates an auxiliary sequence (denoted as $\{\Delta_k\} \subset \mathbb{R}_{>0}$ in [14]) that influences the step acceptance mechanism, which in the context of I-TRACE are the conditions in FDS for determining whether a decrease step has been found or whether the trust-region radius should be expanded or contracted. The role played by this sequence is to ensure that the algorithm converges from remote starting points even if one does not assume that the Hessian function is Lipschitz continuous over the path generated by the algorithm iterates. One could introduce such an auxiliary sequence for I-TRACE that would play this same role.

Algorithm 2.3 I-TRACE.

Parameters: $(\xi_1, \xi_2, \xi_3) \in \mathbb{R}_{>0} \times (0,1) \times \mathbb{R}_{>0}$, $\eta \in (0,1)$, $\underline{\sigma} \in (0,\infty)$, $\overline{\sigma} \in (\underline{\sigma}, \infty)$, $\gamma_C \in (0,1)$, $\gamma_E \in (1,\infty)$, and $\gamma_\lambda \in (1,\infty)$
Input: $x_0 \in \mathbb{R}^n$, $\delta_0 \in (0,\infty)$, $\sigma_0 \in [\underline{\sigma}, \overline{\sigma}]$

for $k = 0, 1, \ldots$ **do**
  $\gamma_{k,0} \leftarrow \|g_k\|$
  $(j, t_{k,j}, \lambda_{k,j}, Q_{k,j}, T_{k,j}, y_{k,j+1}, \gamma_{k,j+1}, \mu_{k,j}) \leftarrow \text{TLTR}(g_k, \gamma_{k,0}, H_k, \delta_k)$
  loop
    $(t_{k,j}, \lambda_{k,j}, \mu_{k,j}, \bar{\delta}_k, \bar{\sigma}_k) \leftarrow \text{FDS}(t_{k,j}, \lambda_{k,j}, Q_{k,j}, T_{k,j}, \gamma_{k,0}, \gamma_{k,j+1}, \mu_{k,j}, \delta_k, \sigma_k)$
    if (2.4) (equivalently, (2.3)) holds **then**
      set $s_k \leftarrow Q_{k,j} t_{k,j}$
      break
    else
      $j \leftarrow j + 1$
      $q_{k,j} \leftarrow \frac{1}{\gamma_{k,j}} y_{k,j}$
      $Q_{k,j} \leftarrow \begin{bmatrix} Q_{k,j-1} & q_{k,j} \end{bmatrix}$
      $\theta_{k,j} \leftarrow q_{k,j}^T H_k q_{k,j}$
      $T_{k,j} \leftarrow \begin{bmatrix} & & 0 \\ & T_{k,j-1} & \vdots \\ & & \gamma_{k,j} \\ 0 & \cdots \quad \gamma_{k,j} & \theta_{k,j} \end{bmatrix}$
      compute $(t_{k,j}, \lambda_{k,j})$ by solving $\mathcal{S}_{k,j}(\delta_k)$ (which gives $s_{k,j} = Q_{k,j} t_{k,j}$)
      $y_{k,j+1} \leftarrow H_k q_{k,j} - \theta_{k,j} q_{k,j} - \gamma_{k,j} q_{k,j-1}$
      $\gamma_{k,j+1} \leftarrow \|y_{k,j+1}\|$
      $\mu_{k,j} \leftarrow \gamma_{k,j+1} |e_{j+1}^T t_{k,j}|$ $(= \|r_{k,j}\|$ where $r_{k,j} = g_k + (H_k + \lambda_{k,j} I) s_{k,j})$
    end if
  end loop
  set $x_{k+1} \leftarrow x_k + s_k$
  set $\delta_{k+1} \leftarrow \max\{\bar{\delta}_k, \gamma_E \|s_k\|\}$
  set $\sigma_{k+1} \leftarrow \bar{\sigma}_k$
end for

However, as mentioned, for the sake of brevity in this paper, we do not analyze the global convergence properties of the algorithm under this more general setting but instead have chosen to include upfront (in Assumption 1.1) a Lipschitz continuity assumption for the Hessian function. In this setting, the auxiliary sequence is not needed to prove the results in this paper, so we have not included it.

3. Convergence and complexity analyses. In this section, we prove convergence and worst-case complexity results for I-TRACE (Algorithm 2.3) under Assumption 1.1. We also add the following assumption, which is reasonable for the purposes of our analysis because if the algorithm reaches an iteration in which the gradient is zero, then (in a finite number of iterations) it satisfies (1.2) for any $\epsilon \in \mathbb{R}_{>0}$.

Assumption 3.1. For all generated $k \in \mathbb{N}$, one finds that $g_k \neq 0$.

We begin by proving preliminary results that show that the algorithm is well posed in the sense that it will generate an infinite sequence of iterates. These results rely heavily on the algorithm's use of the Lanczos process for generating the basis

matrices that define the reduced-space subproblems. We then prove our first main set of results on the algorithm's worst-case complexity properties to approximate first-order stationarity (recall (1.2)). A consequence of these results is that the algorithm converges from remote starting points. Finally, we prove that, as an (inexact) second-order method, I-TRACE can achieve a rate of local convergence comparable to TRACE.

In various parts of our analysis, we refer to results in "standard trust-region theory," such as on the relationship between primal and dual trust-region subproblem solutions. For the sake of brevity, we do not cite particular lemmas for all such results; the reader may refer to textbooks such as [11, 31] for the results that we use.

3.1. Preliminary results. In this subsection, we show that I-TRACE is well posed in the sense that, for all generated $k \in \mathbb{N}$, the **for** loop in TLTR terminates finitely, each call to FDS terminates finitely, and the inner **loop** in I-TRACE terminates finitely, which together show that I-TRACE reaches iteration $k+1$. Inductively, this means that the algorithm generates iterates ad infinitum.

Supposing that I-TRACE has reached iteration $k \in \mathbb{N}$, our first results in this subsection show that the call to TLTR terminates finitely. Our presentation of this subroutine is based on the claim that $\mu_{k,j} = \|r_{k,j}\|$ for all generated j, where for each such j the vector $r_{k,j}$ is the residual corresponding to $s_{k,j} = Q_{k,j}t_{k,j}$ as defined in (2.2a). The following lemma, proved as [20, Theorem 5.1], formalizes this claim.

LEMMA 3.1 ([20, Theorem 5.1]). *For all generated $k \in \mathbb{N}$, the call to* TLTR *yields*

$$r_{k,j} = (H_k + \lambda_{k,j} I)\, Q_{k,j} t_{k,j} + g_k = \gamma_{k,j+1} e_{j+1}^T t_{k,j} q_{k,j+1} \text{ for all generated } j \in \mathbb{N},$$

from which it follows that $\mu_{k,j} \leftarrow \gamma_{k,j+1} |e_{j+1}^T t_{k,j}| = \|r_{k,j}\|$ for all such j.

We now show that any call to TLTR by I-TRACE terminates finitely.

LEMMA 3.2. *For all generated $k \in \mathbb{N}$, the call to* TLTR *yields $(s_{k,j}, \lambda_{k,j})$ satisfying* (2.2) *for all generated $j \in \mathbb{N}$ and terminates finitely; more precisely, the call to* TLTR *terminates in iteration j for some $j \in \{0, \ldots, n-1\}$.*

Proof. Consider arbitrary $k \in \mathbb{N}$ generated by I-TRACE. Since it is based on the Lanczos process, it is well known (see, e.g., [36]) that if TLTR were to continue to iterate, then the dimension of the Krylov subspace $\mathcal{K}_{k,j}$ would increase by 1 whenever j increases by 1 until it reaches some $\bar{j} \in \{0, \ldots, n-1\}$ corresponding to which one finds that

$$\begin{aligned} \gamma_{k,j} &> 0 \text{ for all } j \in \{0, \ldots, \bar{j}\}, \\ \gamma_{k,\bar{j}+1} &= 0, \quad \text{and} \\ \dim(\mathcal{K}_{k,\bar{j}}) &= \dim(\mathcal{K}_{k,\bar{j}+1}) = \bar{j} + 1. \end{aligned} \tag{3.1}$$

Our goal is to show that $(s_{k,j}, \lambda_{k,j})$ satisfies (2.2) for all generated $j \in \{0, \ldots, \bar{j}\}$ and that TLTR terminates by iteration $\bar{j}$ at the latest.

Consider arbitrary generated $j \in \{0, \ldots, \bar{j}\}$. It follows by Lemma 3.1 that (2.2a) holds. In addition, by Lemma 3.1 and the fact that $Q_{k,j}^T q_{k,j+1} = 0$, one finds that

$$r_{k,j}^T s_{k,j} = \left(\gamma_{k,j+1} e_{j+1}^T t_{k,j} q_{k,j+1}\right)^T Q_{k,j} t_{k,j} = 0,$$

meaning that (2.2b) holds. Moreover, since $(t_{k,j}, \lambda_{k,j})$ is a globally optimal solution of $\mathcal{S}_{k,j}(\delta_k)$, it satisfies (2.1b), which, in turn, means that $s_{k,j} \equiv Q_{k,j} t_{k,j}$ and $\lambda_{k,j}$ yield

$$\begin{aligned} s_{k,j}^T(H_k + \lambda_{k,j} I)s_{k,j} &= t_{k,j}^T \left(Q_{k,j}^T H_k Q_{k,j} + \lambda_{k,j} Q_{k,j}^T Q_{k,j}\right) t_{k,j} \\ &= t_{k,j}^T (T_{k,j} + \lambda_{k,j} I) t_{k,j} \geq 0, \end{aligned}$$

meaning that (2.2c) holds. Finally, the fact that $(t_{k,j}, \lambda_{k,j})$ satisfies (2.1c) means that $(s_{k,j}, \lambda_{k,j})$ (satisfying $\|s_{k,j}\| = \|Q_{k,j} t_{k,j}\| = \|t_{k,j}\|$) satisfies (2.2d). Overall, it has been shown that (2.2a)–(2.2d) hold for all $j \in \{0, \ldots, \overline{j}\}$.

To complete the proof, all that remains is to observe that for $j = \overline{j}$, one finds from Lemma 3.1 and (3.1) that $\mu_{k,\overline{j}} = \|r_{k,\overline{j}}\| = 0 \leq \xi \|t_{k,j}\|^2$, meaning that (2.4a) holds. Hence, TLTR would terminate in iteration $\overline{j}$ if it does not terminate earlier. ∎

Our next set of results show that any call to FDS terminates finitely. First, it is clear that each line of FDS is well posed—since each line involves either a straightforward computation or the computation of a solution of a well-defined subproblem—with the only possible exception being the computation requested in lines 15–16 of FDS. The fact that this computation is well posed is proved in the following lemma.

LEMMA 3.3. *For all generated $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ such that lines* 15–16 *of* FDS *are reached, the required computation is well posed.*

Proof. Consider arbitrary $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ such that lines 15–16 of FDS are reached. By construction, one must have $\lambda_{k,j,l} < \underline{\sigma} \|t_{k,j,l}\|$, $\lambda_{k,j,l} < \hat{\lambda}_{k,j,l+1}$, and $\overline{\sigma} \|\hat{t}_{k,j,l+1}\| < \hat{\lambda}_{k,j,l+1}$, where $(t_{k,j,l}, \lambda_{k,j,l})$ solves $\mathcal{S}_{k,j}(\delta_{k,j,l})$, and $\hat{t}_{k,j,l+1}$ solves $\mathcal{R}_{k,j}(\hat{\lambda}_{k,j,l+1})$. It follows by standard trust-region theory that the ratio function $\phi : [\lambda_{k,j,l}, \infty] \to \overline{\mathbb{R}}$ defined by $\phi(\lambda) = \lambda / \|t_{k,j}(\lambda)\|$ with $t_{k,j}(\lambda)$ defined as the solution of $\mathcal{R}_{k,j}(\lambda)$ is monotonically increasing. Therefore, along with the aforementioned inequalities, it follows that there exists $\bar{\lambda}_{k,j,l+1}$ such that the solution $\bar{t}_{k,j,l+1}$ of $\mathcal{R}_{k,j}(\bar{\lambda}_{k,j,l+1})$ yields $\underline{\sigma} < \frac{\bar{\lambda}_{k,j,l+1}}{\|\bar{t}_{k,j,l+1}\|} < \overline{\sigma}$, as claimed. ∎

Now, having shown that each line of FDS is well posed, we proceed to show that the **for** loop of the subroutine terminates finitely. We next show that, as is comparable in TRACE, for all generated $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ the pair $(t_{k,j,l}, \lambda_{k,j,l})$ is a primal-dual globally optimal solution of $\mathcal{S}_{k,j}(\delta_{k,j,l})$. We also show that each such pair corresponds to a pair satisfying (2.2), although (2.4) might not hold.

LEMMA 3.4. *For all generated $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$, one finds that the pair $(t_{k,j,l}, \lambda_{k,j,l})$ satisfies* (2.1)*, and the pair $(Q_{k,j} t_{k,j,l}, \lambda_{k,j,l})$ (with $Q_{k,j} t_{k,j,l}$ in place of $s_{k,j}$ and $\lambda_{k,j,l}$ in place of $\lambda_{k,j}$) satisfies* (2.2)*.*

Proof. Consider arbitrary generated $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$. The desired conclusion about $(t_{k,j,l}, \lambda_{k,j,l})$ follows since it is obtained either directly by solving $\mathcal{S}_{k,j}(\delta_{k,j,l})$ or by solving $\mathcal{R}_{k,j}(\lambda_{k,j,l})$ to obtain $t_{k,j,l}$ and then subsequently setting $\delta_{k,j,l} \leftarrow \|t_{k,j,l}\|$, which again means that $(t_{k,j,l}, \lambda_{k,j,l})$ solves $\mathcal{S}_{k,j}(\delta_{k,j,l})$. As for the second desired conclusion, first note from Lemma 3.2 that it holds for $l = 0$. Then, using the same logic as in the proof of the first desired conclusion, observe for the index l of interest that the pair $(t_{k,j,l}, \lambda_{k,j,l})$ satisfies (2.1) with the same $(Q_{k,j}, T_{k,j}, \gamma_{k,0}, \gamma_{k,j+1})$ as for $l = 0$; all that has changed between $l = 0$ and the value of l of interest is the trust-region radius. Hence, the desired conclusion follows using the same logic as in the proof of Lemma 3.2, except that (2.4a) (hence, (2.4)) might no longer be satisfied. ∎

Our next lemma shows that each trial step is nonzero, the proof of which is merely an adaptation of [14, Lemma 3.2] to the setting of I-TRACE.

LEMMA 3.5. *For all generated $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$, one finds $\|t_{k,j,l}\| > 0$.*

Proof. Consider arbitrary generated $(k,j,l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$. By Lemma 3.4, $(t_{k,j,l}, \lambda_{k,j,l})$ is a primal-dual globally optimal solution of $\mathcal{S}_{k,j}(\delta_{k,j,l})$. If $T_{k,j} = 0$, then (2.1a) implies $\gamma_{k,0}e_1 + \lambda_{k,j,l}t_{k,j,l} = 0$, which, since $\gamma_{k,0} = \|g_k\| \neq 0$ (under Assumption 3.1), means that $\lambda_{k,j,l} \neq 0$ and $t_{k,j,l} \neq 0$, from which the desired conclusion holds. On the other hand, if $T_{k,j} \neq 0$, then there are two cases. If $\|t_{k,j,l}\| = \delta_{k,j,l}$, then since $\delta_{k,j,l} > 0$ by construction of the algorithm the desired conclusion holds; otherwise, $\|t_{k,j,l}\| < \delta_{k,j,l}$, in which case (2.1a) and (2.1c) imply $\gamma_{k,0}e_1 + T_{k,j}t_{k,j,l} = 0$, from which it follows (under Assumption 3.1) that $\|t_{k,j,l}\| \geq \gamma_{k,0}/\|T_{k,j}\| > 0$. ∎

We now show that if a computed dual subproblem solution is sufficiently large relative to the norm of the corresponding primal subproblem solution, then the trust-region constraint must be active, and sufficient decrease is offered.

LEMMA 3.6. *For all generated $(k,j,l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$, if $(t_{k,j,l}, \lambda_{k,j,l})$ yields*

$$\lambda_{k,j,l} \geq (H_{\text{Lip}} + 2\eta)\|t_{k,j,l}\|, \tag{3.2}$$

then $\|t_{k,j,l}\| = \delta_{k,j,l}$ and $\rho_k(Q_{k,j}t_{k,j,l}) \geq \eta$.

Proof. Consider arbitrary generated $(k,j,l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$. By Lemma 3.5, one finds that $\|t_{k,j,l}\| > 0$, implying that $\lambda_{k,j,l} \geq (H_{\text{Lip}} + 2\eta)\|t_{k,j,l}\| > 0$. Hence, by Lemma 3.4 and (2.1c), one finds that $\|t_{k,j,l}\| = \delta_{k,j,l}$, which is the first desired conclusion. Now observe that Assumption 1.1 and Taylor's theorem imply that there exists a point $\overline{x}_k$ on the line segment $[x_k, x_k + Q_{k,j}t_{k,j,l}]$ such that

$$\begin{aligned} &m_k(Q_{k,j}t_{k,j,l}) - f(x_k + Q_{k,j}t_{k,j,l}) \\ &\quad = f_k + g_k^T Q_{k,j}t_{k,j,l} + \tfrac{1}{2}t_{k,j,l}^T Q_{k,j}^T H_k Q_{k,j}t_{k,j,l} - f(x_k + Q_{k,j}t_{k,j,l}) \\ &\quad = \tfrac{1}{2}t_{k,j,l}^T Q_{k,j}^T (H_k - H(\overline{x}_k))Q_{k,j}t_{k,j,l} \geq -\tfrac{1}{2}H_{\text{Lip}}\|t_{k,j,l}\|^3. \end{aligned}$$

On the other hand, since $(Q_{k,j}t_{k,j,l}, \lambda_{k,j,l})$ satisfies (2.2a)–(2.2c) by Lemma 3.4, it follows that for some $r_{k,j,l} \in \mathbb{R}^n$ such that $t_{k,j,l}^T Q_{k,j}^T r_{k,j,l} \leq 0$ one finds

$$\begin{aligned} f_k - m_k(Q_{k,j}t_{k,j,l}) &= -g_k^T Q_{k,j}t_{k,j,l} - t_{k,j,l}^T Q_{k,j}^T H_k Q_{k,j}t_{k,j,l} + \tfrac{1}{2}t_{k,j,l}^T Q_{k,j}^T H_k Q_{k,j}t_{k,j,l} \\ &= -t_{k,j,l}^T Q_{k,j}^T (g_k + H_k Q_{k,j}t_{k,j,l}) + \tfrac{1}{2}t_{k,j,l}^T Q_{k,j}^T H_k Q_{k,j}t_{k,j,l} \\ &= -t_{k,j,l}^T Q_{k,j}^T (r_{k,j,l} - \lambda_{k,j,l}Q_{k,j}t_{k,j,l}) + \tfrac{1}{2}t_{k,j,l}^T Q_{k,j}^T H_k Q_{k,j}t_{k,j,l} \\ &= -t_{k,j,l}^T Q_{k,j}^T r_{k,j,l} + \tfrac{1}{2}t_{k,j,l}^T Q_{k,j}^T (H_k + \lambda_{k,j,l}I)Q_{k,j}t_{k,j,l} + \tfrac{1}{2}\lambda_{k,j,l}\|t_{k,j,l}\|^2 \\ &\geq \tfrac{1}{2}\lambda_{k,j,l}\|t_{k,j,l}\|^2. \end{aligned}$$

Therefore, if (3.2) holds, then one finds that

$$\begin{aligned} \rho_k(Q_{k,j}t_{k,j,l}) &= \frac{f_k - f(x_k + Q_{k,j}t_{k,j,l})}{\|Q_{k,j}t_{k,j,l}\|^3} \\ &= \frac{f_k - m_k(Q_{k,j}t_{k,j,l})}{\|t_{k,j,l}\|^3} + \frac{m_k(Q_{k,j}t_{k,j,l}) - f(x_k + Q_{k,j}t_{k,j,l})}{\|t_{k,j,l}\|^3} \\ &\geq \frac{-\frac{1}{2}H_{\text{Lip}}\|t_{k,j,l}\|^3 + \frac{1}{2}\lambda_{k,j,l}\|t_{k,j,l}\|^2}{\|t_{k,j,l}\|^3} \geq -\tfrac{1}{2}H_{\text{Lip}} + \tfrac{1}{2}(H_{\text{Lip}} + 2\eta) = \eta, \end{aligned}$$

as desired. ∎

We now partition the set of indices generated within any call to FDS and proceed to show that the number of each type of iteration is finite. In particular, let us define, for all generated $(k,j) \in \mathbb{N} \times \mathbb{N}$ such that FDS is called, the index sets

$$\mathcal{A}_{k,j} := \left\{ l \in \mathbb{N} : \text{index } l \text{ is reached and } \rho_k(Q_{k,j} t_{k,j,l}) \geq \eta \text{ and } \frac{\lambda_{k,j,l}}{\|t_{k,j,l}\|} \leq \sigma_{k,j,l} \right\},$$
$$\mathcal{E}_{k,j} := \left\{ l \in \mathbb{N} : \text{index } l \text{ is reached and } \rho_k(Q_{k,j} t_{k,j,l}) \geq \eta \text{ and } \frac{\lambda_{k,j,l}}{\|t_{k,j,l}\|} > \sigma_{k,j,l} \right\},$$
$$\text{and } \mathcal{C}_{k,j} := \{ l \in \mathbb{N} : \text{index } l \text{ is reached and } \rho_k(Q_{k,j} t_{k,j,l}) < \eta \},$$

which, respectively, represent the indices corresponding to *accepted, expansion,* and *contraction* steps in the call to FDS corresponding to (k, j). It follows trivially by construction of FDS that $|\mathcal{A}_{k,j}| \leq 1$. Our next lemma shows that if an expansion or contraction step occurs, then the subsequent step cannot be an expansion step. This is a critical feature of TRACE as well; see [14, Lemma 3.7].

LEMMA 3.7. *For all generated* $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$, *if* $l \in \mathcal{E}_{k,j} \cup \mathcal{C}_{k,j}$, *then* $(k, j, l+1)$ *is generated, and* $(l+1) \notin \mathcal{E}_{k,j}$.

Proof. Consider arbitrary generated $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ such that $l \in \mathcal{E}_{k,j} \cup \mathcal{C}_{k,j}$. It follows by construction of FDS that iteration $l+1$ will be reached.

Case 1: $\lambda_{k,j,l+1} = 0$. Since $\sigma_{k,j,0} > 0$ by construction of I-TRACE, it follows by lines 7 and 29 in FDS that $\sigma_{k,j,l+1} > 0$. Hence, one finds that $\sigma_{k,j,l+1} > 0 = \frac{\lambda_{k,j,l+1}}{\|t_{k,j,l}\|}$, from which it follows that $(t+1) \notin \mathcal{E}_{k,j}$.

Case 2: $\lambda_{k,j,l+1} > 0$. By Lemma 3.4 and (2.1c), it follows that $\|t_{k,j,l+1}\| = \delta_{k,j,l+1}$. If $l \in \mathcal{E}_{k,j}$, then one finds that $\|t_{k,j,l+1}\| = \delta_{k,j,l+1} = \frac{\lambda_{k,j,l}}{\sigma_{k,j,l}} > \|t_{k,j,l}\| = \delta_{k,j,l}$, so by standard trust-region theory one finds $\lambda_{k,j,l+1} < \lambda_{k,j,l}$. Hence,

$$\frac{\lambda_{k,j,l+1}}{\|t_{k,j,l+1}\|} \leq \frac{\lambda_{k,j,l} \sigma_{k,j,l}}{\lambda_{k,j,l}} = \sigma_{k,j,l} = \sigma_{k,j,l+1},$$

from which it follows that $(t+1) \notin \mathcal{E}_{k,j}$. On the other hand, if $l \in \mathcal{C}_{k,j}$, then line 29 ensures that $\sigma_{k,j,l+1} \geq \frac{\lambda_{k,j,l+1}}{\|t_{k,j,l+1}\|}$, so $(l+1) \notin \mathcal{E}_{k,j}$.

The conclusion follows by combining the results of the two cases. ∎

An immediate consequence of the previous lemma is that the number of expansion steps in any call to FDS is limited by one.

LEMMA 3.8. *For all generated* $(k, j) \in \mathbb{N} \times \mathbb{N}$ *such that* FDS *is called,* $|\mathcal{E}_{k,j}| \leq 1$.

Proof. Consider arbitrary generated $(k, j) \in \mathbb{N} \times \mathbb{N}$ such that FDS is called. If $|\mathcal{E}_{k,j}| = 0$, then there is nothing left to prove. Otherwise, for some smallest generated $l \in \mathbb{N}$ one finds that $l \in \mathcal{E}_{k,j}$. It then follows by induction that $|\mathcal{E}_{k,j}| = 1$. This can be seen to follow from the fact that, since $l \in \mathcal{E}_{k,j}$, Lemma 3.7 implies that $(l+1) \in \mathcal{A}_{k,j} \cup \mathcal{C}_{k,j}$. If $(l+1) \in \mathcal{A}_{k,j}$, then FDS terminates and $|\mathcal{E}_{k,j}| = 1$, while if $(l+1) \in \mathcal{C}_{k,j}$, then Lemma 3.7 implies that $(l+2) \in \mathcal{A}_{k,j} \cup \mathcal{C}_{k,j}$. This argument shows inductively that $|\mathcal{E}_{k,j}| = 1$, as claimed. ∎

All that remains in order to prove that any call to FDS terminates finitely is to prove that the number of contraction steps is finite. Toward this end, we now prove that as the result of any contraction step, the trust-region radius is decreased, and the dual subproblem solution does not decrease. The proof of the following lemma is essentially the same as that for [14, Lemma 3.4], but we provide it for completeness.

LEMMA 3.9. *For all generated* $(k, j, l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$, *if* $l \in \mathcal{C}_{k,j}$, *then* $(k, j, l+1)$ *is generated,* $\delta_{k,j,l+1} < \delta_{k,j,l}$, *and* $\lambda_{k,j,l+1} \geq \lambda_{k,j,l}$.

Proof. Consider arbitrary generated $(k,j,l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ such that $l \in \mathcal{C}_{k,j}$. If line 13, 17, or 23 is reached, then $\delta_{k,j,l+1} \leftarrow \|t_{k,j,l+1}\|$ where $t_{k,j,l+1}$ solves $\mathcal{R}_{k,j}(\lambda)$ for $\lambda > \lambda_{k,j,l}$. Since $\lambda > \lambda_{k,j,l}$, it follows by standard trust-region theory that

$$\delta_{k,j,l+1} \leftarrow \|t_{k,j,l+1}\| < \|t_{k,j,l}\| \leq \delta_{k,j,l} \quad \text{and} \quad \lambda_{k,j,l+1} = \lambda > \lambda_{k,j,l}.$$

The only other possibility is that line 25 is reached, in which case one finds $\delta_{k,j,l+1} \leftarrow \gamma_C \|t_{k,j,l}\| < \delta_{k,j,l}$, which by standard trust-region theory implies $\lambda_{k,j,l+1} \geq \lambda_{k,j,l}$. ∎

We are now prepared to prove that, in any call to FDS, the number of contraction steps is finite, which, along with previous results, shows that FDS terminates finitely.

LEMMA 3.10. *For all generated $(k,j) \in \mathbb{N} \times \mathbb{N}$ such that* FDS *is called, the call to* FDS *terminates finitely.*

Proof. Consider arbitrary generated $(k,j) \in \mathbb{N} \times \mathbb{N}$ such that FDS is called. As already observed, by construction of FDS, it follows that $|\mathcal{A}_{k,j}| \leq 1$. Moreover, by Lemma 3.8, it follows that $|\mathcal{E}_{k,j}| \leq 1$. Hence, it remains to prove that $|\mathcal{C}_{k,j}| < \infty$.

In order to derive a contradiction, suppose that $|\mathcal{C}_{k,j}| = \infty$, which, along with Lemma 3.8, means that $l \in \mathcal{C}_{k,j}$ for all sufficiently large $l \in \mathbb{N}$. Indeed, we may assume, without loss of generality, that $\mathcal{C}_{k,j} = \mathbb{N}$. Our goal now is to show—using the arguments of [14, Lemma 3.9]—that $\{\delta_{k,j,l}\}_{l\in\mathbb{N}} \to 0$ and $\{\lambda_{k,j,l}\}_{l\in\mathbb{N}} \to \infty$. By Lemma 3.9 and the fact that $\{\delta_{k,j,l}\}_{l\in\mathbb{N}} \subset \mathbb{R}_{>0}$ by construction, it follows that $\{\delta_{k,j,l}\}_{l\in\mathbb{N}}$ converges. If line 25 is reached infinitely often, then $\{\delta_{k,j,l}\}_{l\in\mathbb{N}} \to 0$ and, by standard trust-region theory, $\{\lambda_{k,j,l}\}_{l\in\mathbb{N}} \to \infty$, as desired. Hence, we may assume that line 25 is reached only a finite number of times. Let us now prove that we may also proceed under the assumption that line 17 is only reached a finite number of times. Suppose that for some $l \in \mathbb{N}$ one finds that line 17 is reached, in which case the algorithm sets $(t_{k,j,l+1}, \lambda_{k,j,l+1})$ such that during iteration $(l+1) \in \mathcal{C}_{k,j}$ the condition in line 9 will test false, meaning that the algorithm will proceed to line 20 in iteration $l+1$. Since, by Lemma 3.9, $\{\delta_{k,j,l}\}_{l\in\mathbb{N}}$ is monotonically decreasing and $\{\lambda_{k,j,l}\}_{l\in\mathbb{N}}$ is monotonically nondecreasing, it follows that $\{\lambda_{k,j,l}/\|t_{k,j,l}\|\}_{l\in\mathbb{N}}$ is monotonically increasing, which means that the condition in line 9 will test false in all subsequent iterations, meaning that line 17 is only reached a finite number of times, as claimed. All that remains in order to prove $\{\delta_{k,j,l}\}_{l\in\mathbb{N}} \to 0$ and $\{\lambda_{k,j,l}\}_{l\in\mathbb{N}} \to \infty$ is to show that these limits hold under the assumption that line 13 or 23 is reached for all $l \geq \bar{l}$ for some $\bar{l} \in \mathbb{N}$. Under this assumption, one finds that

$$\lambda_{k,j,l+1} \geq \min\{\lambda_{k,j,l} + (\underline{\sigma}\gamma_{k,0})^{1/2}, \gamma_\lambda \lambda_k\} \quad \text{for all} \quad l \geq \bar{l} + 1,$$

which implies that, in fact, $\{\lambda_{k,j,l}\}_{l\in\mathbb{N}} \to \infty$. According to standard trust-region theory, this shows that $\{\delta_{k,j,l}\}_{l\in\mathbb{N}} \to 0$, as desired.

Since it has been shown that $\mathcal{C}_{k,j} = \mathbb{N}$ implies that one has $\{\delta_{k,j,l}\}_{l\in\mathbb{N}} \to 0$ and $\{\lambda_{k,j,l}\}_{l\in\mathbb{N}} \to \infty$, one may now conclude from Lemma 3.6 that $l \notin \mathcal{C}_{k,j}$ for some sufficiently large $l \in \mathbb{N}$, which is a contradiction to the fact that $\mathcal{C}_{k,j} = \mathbb{N}$. ∎

We may now prove our concluding result of this subsection.

LEMMA 3.11. I-TRACE *generates an infinite sequence of iterates, where for all generated $(k,j) \in \mathbb{N} \times \mathbb{N}$ one finds that $j \in \{0, \ldots, n-1\}$.*

Proof. The result follows by induction. Supposing that I-TRACE reaches iteration $k \in \mathbb{N}$, it follows from Lemma 3.2 that the call to TLTR terminates finitely with $j \leq n-1$, and it follows from Lemma 3.10 that any call to FDS

terminates finitely. Hence, all that remains it to prove that the **loop** in I-TRACE terminates finitely, since this means that I-TRACE reaches iteration $k+1$. This follows using the same argument as in the proof of Lemma 3.2, because if j reaches $\bar{j} \in \{0, \ldots, n-1\}$ such that (3.1) holds, the output from FDS yields $\mu_{k,\bar{j}} = 0$, in which case the **loop** will terminate. ∎

3.2. Worst-case complexity. Our purpose in this subsection is to prove worst-case complexity bounds pertaining to I-TRACE's pursuit of ϵ-stationarity. In fact, in this subsection we show upper bounds on the total numbers of iterations, function evaluations, derivative evaluations, and Hessian-vector products that I-TRACE may perform at iterates at which, for arbitrary $\epsilon \in (0,1)$, the bound (1.2) does not hold. Since this iteration bound holds for arbitrary $\epsilon \in (0,1)$, it follows immediately that I-TRACE converges toward first-order stationarity in the limit, i.e., $\{\|g_k\|\} \to 0$.

Our first lemma of this subsection shows that, as in TRACE, a contraction step causes the ratio of the dual subproblem solution to the norm of the primal subproblem solution to obey certain iteration-dependent and uniform bounds.

LEMMA 3.12. *For all generated $(k,j,l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$, if $l \in \mathcal{C}_{k,j}$, then*

$$\underline{\sigma} \leq \frac{\lambda_{k,j,l+1}}{\|t_{k,j,l+1}\|} \leq \max\left\{\overline{\sigma}, \left(\frac{\gamma_\lambda}{\gamma_C}\right) \frac{\lambda_{k,j,l}}{\|t_{k,j,l}\|}\right\} \leq \max\left\{\overline{\sigma}, \left(\frac{\gamma_\lambda}{\gamma_C}\right)(H_{\text{Lip}} + 2\eta)\right\}.$$

If, in addition, $\lambda_{k,j,l} \geq \underline{\sigma}\|t_{k,j,l}\|$, then

$$\frac{\lambda_{k,j,l+1}}{\|t_{k,j,l+1}\|} \geq \min\left\{\gamma_\lambda, \frac{1}{\gamma_C}\right\} \frac{\lambda_{k,j,l}}{\|t_{k,j,l}\|}.$$

Proof. The proof of the first two desired inequalities follows using the same reasoning as in the proof of [14, Lemma 3.17], the details of which we omit for the sake of brevity. The next desired inequality follows from Lemma 3.6 and the fact that $l \in \mathcal{C}_{k,j}$ only if $\lambda_{k,j,l} < (H_{\text{Lip}} + 2\eta)\|t_{k,j,l}\|$. Finally, under the additional condition that $\lambda_{k,j,l} \geq \underline{\sigma}\|t_{k,j,l}\|$, the final desired conclusion follows using the same reasoning as in the proof of [14, Lemma 3.23], where again we omit the details for brevity. ∎

We now use the previous lemma to prove a critical upper bound.

LEMMA 3.13. *Defining*

$$\sigma_{\max} := \max\left\{\sigma_0, \overline{\sigma}, \left(\frac{\gamma_\lambda}{\gamma_C}\right)(H_{\text{Lip}} + 2\eta)\right\} > 0,$$

it follows for all generated $(k,j,l) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ that $\sigma_{k,j,l} \leq \sigma_{\max}$.

Proof. We prove the result by induction. As a base case, consider $k = 0$ and the corresponding smallest $j \in \mathbb{N}$ such that FDS is called. Given $k = 0$ and such a $j \in \mathbb{N}$, the call to FDS initializes $\sigma_{0,j,0} = \sigma_0 \leq \sigma_{\max}$. Now suppose that for arbitrary generated $k \in \mathbb{N}$ and the corresponding smallest $j \in \mathbb{N}$ such that FDS is called, one finds for generated $l \in \mathbb{N}$ that $\sigma_{k,j,l} \leq \sigma_{\max}$. If $l \in \mathcal{E}_{k,j}$, then by line 7 of FDS one finds that $\sigma_{k,j,l+1} = \sigma_{k,j,l} \leq \sigma_{\max}$. If $l \in \mathcal{C}_{k,j}$, then by Lemma 3.12 and line 29 one finds that

$$\sigma_{k,j,l+1} \leftarrow \max\left\{\sigma_{k,j,l}, \frac{\lambda_{k,j,l+1}}{\|t_{k,j,l+1}\|}\right\} \leq \max\left\{\sigma_{k,j,l}, \overline{\sigma}, \left(\frac{\gamma_\lambda}{\gamma_C}\right)(H_{\text{Lip}} + 2\eta)\right\} \leq \sigma_{\max}.$$

Finally, if $l \in \mathcal{A}_{k,j}$, then either (i) (2.4) is satisfied, I-TRACE proceeds to (outer) iteration $k+1$, and for some smallest corresponding $j \in \mathbb{N}$ such that FDS is called, one

finds $\sigma_{k+1,j,0} = \sigma_{k,j,l} \leq \sigma_{\max}$; or (ii) (2.4) is not satisfied, I-TRACE proceeds to (inner) iteration $j+1$, and $\sigma_{k,j+1,0} = \sigma_{k,j,0} \leq \sigma_{\max}$. Overall, in all cases, the procedures of I-TRACE and FDS ensure that the desired conclusion holds. ∎

We have proved in Lemma 3.11 that I-TRACE generates an infinite sequence of iterates, meaning that it generates an infinite sequence of steps $\{s_k\}$. For our next result, we prove a critical relationship between the norm of each step and the norm of the gradient of the objective function at the subsequent iterate. Such a relationship is critical for all of the optimal-complexity methods mentioned in section 1.2.

LEMMA 3.14. *For all $k \in \mathbb{N}$, the step s_k satisfies*

$$\|s_k\| \geq \left(\frac{1-\xi_2}{\frac{1}{2}H_{\mathrm{Lip}} + \sigma_{\max} + \max\{\xi_1, \xi_2 g_{\mathrm{Lip}}\}} \right)^{1/2} \|g_{k+1}\|^{1/2}.$$

Proof. Consider arbitrary $k \in \mathbb{N}$. By construction of FDS and I-TRACE and by Lemma 3.13, one has at line 8 of I-TRACE (with $\lambda_k \leftarrow \lambda_{k,j}$) that $\lambda_k \leq \sigma_{\max}\|s_k\|$ and

$$\begin{aligned} \text{either} \quad & \|g_k + (H_k + \lambda_k I)s_k\| \leq \xi_1 \|s_k\|^2 \\ \text{or} \quad & \|g_k + (H_k + \lambda_k I)s_k\| \leq \xi_2 \min\{1, \|s_k\|\}\|g_k\|. \end{aligned}$$

Under Assumption 1.1, one finds that

$$\|g_k\| \leq \|g_{k+1}\| + \|g_{k+1} - g_k\| \leq \|g_{k+1}\| + g_{\mathrm{Lip}}\|s_k\|;$$

hence, either $\|g_k + (H_k + \lambda_k I)s_k\| \leq \xi_1\|s_k\|^2 \leq \xi_1\|s_k\|^2 + \xi_2\|g_{k+1}\|$ or

$$\begin{aligned} \|g_k + (H_k + \lambda_k I)s_k\| &\leq \xi_2 \min\{1, \|s_k\|\}(\|g_{k+1}\| + g_{\mathrm{Lip}}\|s_k\|) \\ &\leq \xi_2(\|g_{k+1}\| + g_{\mathrm{Lip}}\|s_k\|^2). \end{aligned}$$

Overall, since

$$\begin{aligned} \|g_{k+1}\| &= \|g(x_k + s_k) - (g_k + (H_k + \lambda_k I)s_k) + (g_k + (H_k + \lambda_k I)s_k)\| \\ &\leq \|g(x_k + s_k) - g_k - H_k s_k\| + \lambda_k\|s_k\| + \|g_k + (H_k + \lambda_k I)s_k\|, \end{aligned}$$

it follows from above that under Assumption 1.1 one has

$$\begin{aligned} &(1-\xi_2)\|g_{k+1}\| \\ &\quad \leq \left\| \int_0^1 (H(x_k + \tau s_k) - H_k)s_k \mathrm{d}\tau \right\| + \sigma_{\max}\|s_k\|^2 + \max\{\xi_1, \xi_2 g_{\mathrm{Lip}}\}\|s_k\|^2 \\ &\quad \leq \left(\int_0^1 \|(H(x_k + \tau s_k) - H_k\| \mathrm{d}\tau \right) \|s_k\| + \sigma_{\max}\|s_k\|^2 + \max\{\xi_1, \xi_2 g_{\mathrm{Lip}}\}\|s_k\|^2 \\ &\quad \leq \left(\int_0^1 \tau \mathrm{d}\tau \right) H_{\mathrm{Lip}}\|s_k\|^2 + \sigma_{\max}\|s_k\|^2 + \max\{\xi_1, \xi_2 g_{\mathrm{Lip}}\}\|s_k\|^2 \\ &\quad = \tfrac{1}{2}H_{\mathrm{Lip}}\|s_k\|^2 + \sigma_{\max}\|s_k\|^2 + \max\{\xi_1, \xi_2 g_{\mathrm{Lip}}\}\|s_k\|^2 \\ &\quad = \left(\tfrac{1}{2}H_{\mathrm{Lip}} + \sigma_{\max} + \max\{\xi_1, \xi_2 g_{\mathrm{Lip}}\}\right)\|s_k\|^2, \end{aligned}$$

which after rearrangement leads to the desired conclusion. ∎

It follows from the preceding lemma that the total number of outer iterations that can be performed by I-TRACE at iterates at which the norm of the gradient is above $\epsilon \in (0,1)$ is $\mathcal{O}(\epsilon^{-3/2})$, which, in turn, means that the total number of gradient evaluations at such iterates is also $\mathcal{O}(\epsilon^{-3/2})$. This is formalized in our first theorem.

THEOREM 3.15. *For arbitrary $\epsilon \in (0,1)$, define for* I-TRACE *the index set*

$$\mathcal{K}(\epsilon) := \{k \in \mathbb{N} : \|g_k\| > \epsilon\}.$$

The total number of elements of $\mathcal{K}(\epsilon)$ is at most

$$K(\epsilon) := 1 + \left\lfloor \left(\frac{(f_0 - f_{\inf})(\frac{1}{2}H_{\text{Lip}} + \sigma_{\max} + \max\{\xi_1, \xi_2 g_{\text{Lip}}\})^{3/2}}{\eta(1-\xi_2)^{3/2}} \right) \epsilon^{-3/2} \right\rfloor.$$

Hence, the total number of "outer" iterations and gradient evaluations performed at iterates that are not ϵ-stationary are each $\mathcal{O}(\epsilon^{-3/2})$ for both.

Proof. By the design of I-TRACE and by Lemma 3.14, it follows for all $k \in \mathbb{N}\backslash\{0\}$ that

$$f_{k-1} - f_k \geq \eta \|s_{k-1}\|^3 \geq \eta \left(\frac{1-\xi_2}{\frac{1}{2}H_{\text{Lip}} + \sigma_{\max} + \max\{\xi_1, \xi_2 g_{\text{Lip}}\}} \right)^{3/2} \|g_k\|^{3/2}.$$

Since f is bounded below under Assumption 1.1 and, by construction, I-TRACE ensures that $\{f_k\}$ is monotonically nonincreasing, it follows from this string of inequalities that $|\mathcal{K}(\epsilon)| < \infty$. Therefore, letting $K_\epsilon \in \mathbb{N}$ denote the largest index in $\mathcal{K}(\epsilon)$ and summing the prior inequality through iteration K_ϵ under Assumption 1.1 yields

$$\begin{aligned} f_0 - f_{\inf} \geq f_0 - f_{K_\epsilon} &= \sum_{k=1}^{K_\epsilon} (f_{k-1} - f_k) \\ &\geq \sum_{k \in \mathcal{K}(\epsilon)} \eta \left(\frac{1-\xi_2}{\frac{1}{2}H_{\text{Lip}} + \sigma_{\max} + \max\{\xi_1, \xi_2 g_{\text{Lip}}\}} \right)^{3/2} \|g_k\|^{3/2} \\ &\geq |\mathcal{K}(\epsilon)| \eta \left(\frac{1-\xi_2}{\frac{1}{2}H_{\text{Lip}} + \sigma_{\max} + \max\{\xi_1, \xi_2 g_{\text{Lip}}\}} \right)^{3/2} \epsilon^{3/2}. \end{aligned}$$

After rearrangement and accounting for iteration $k = 0$, the conclusion follows. ∎

Our goal now is to account first for Hessian-vector products and then for function evaluations. The former occur by line 5 of TLTR and line 13 of I-TRACE, and the latter occur by line 2 in FDS. (Hessian-vector products also appear in line 11 of TLTR and line 16 of I-TRACE, but since these involve the same products as needed in lines 5 and 13, respectively, one does not need to account for these products as well. The products can be stored when first computed and reused as needed.) Our analysis here borrows from the residual analysis from [23]. Importantly, in our analysis of I-TRACE and its pursuit of (first-order) ϵ-stationarity, we are able to make use of the analysis from [23] without having to deal with the so-called *hard case* when solving trust-region subproblems. This follows from the fact that the worst-case complexity properties for which I-TRACE has been designed are of the type described in [23], namely, that do not necessitate approximately globally optimal solutions of the arising subproblems. Indeed, as can be seen in the proof of Lemma 3.14 above, finding subproblem solutions with residuals that are sufficiently small is all that is needed for our purposes.

Following [23, section 3.2], we note the following.

LEMMA 3.16. *For all generated $(k,j) \in \mathbb{N} \times \mathbb{N}$, one finds $T_{k,j} + \lambda_{k,j} I \succ 0$.*

Proof. For arbitrary generated $(k,j) \in \mathbb{N} \times \mathbb{N}$, the conclusion is well known as described in [23, section 3.2], where it is important to note that, by construction and Lemma 3.4, the real number $\lambda_{k,j} \in \mathbb{R}_{\geq 0}$ corresponds to a globally optimal solution of $\mathcal{S}_{k,j}(\delta)$ for some $\delta \in \mathbb{R}_{\geq 0}$ (either $\delta \equiv \delta_k$ or $\delta \equiv \bar{\delta}_k$ from FDS). ∎

For each generated $(k,j) \in \mathbb{N} \times \mathbb{N}$, let us write the spectral decomposition $T_{k,j} = V_{k,j}\Omega_{k,j}V_{k,j}^T$, where $V_{k,j} \in \mathbb{R}^{(j+1)\times(j+1)}$ is an orthonormal matrix of eigenvectors, and $\Omega_{k,j} \in \mathbb{R}^{(j+1)\times(j+1)}$ is a diagonal matrix of eigenvalues, denoted by $\{\omega_{k,j}^{(0)}, \ldots, \omega_{k,j}^{(j)}\}$ and ordered such that $\omega_{k,j}^{(0)} \leq \cdots \leq \omega_{k,j}^{(j)}$. For all generated $(k,j) \in \mathbb{N} \times \mathbb{N}$, let us denote the spectral condition number of $T_{k,j} + \lambda_{k,j} I \succ 0$ (recall Lemma 3.16) as

$$\kappa_{k,j} := \frac{\omega_{k,j}^{(j)} + \lambda_{k,j}}{\omega_{k,j}^{(0)} + \lambda_{k,j}} \in \mathbb{R}_{>0}.$$

Our next result provides an upper bound on the residual defined in (2.2).

LEMMA 3.17. *For all generated $(k,j) \in \mathbb{N} \times \mathbb{N}$, one has that*

$$\|r_{k,j}\| \leq \left(\frac{2\|g_k\| H_{\max}\kappa_{k,j}}{\omega_{k,j}^{(j)} + \lambda_{k,j}}\right)\left(\frac{\sqrt{\kappa_{k,j}} - 1}{\sqrt{\kappa_{k,j}} + 1}\right)^j.$$

Proof. Consider arbitrary generated $(k,j) \in \mathbb{N} \times \mathbb{N}$. By construction of TLTR and by Lemma 3.4, it follows that $(t_{k,j}, \lambda_{k,j})$ satisfies (2.1), and $(s_{k,j}, \lambda_{k,j})$ with $s_{k,j} = Q_{k,j}t_{k,j}$ satisfies (2.2). Hence, by [23, Theorem 3.4], it follows that

$$\|r_{k,j}\| \leq \left(\frac{2\|g_k\| \gamma_{k,j+1}\kappa_{k,j}}{\omega_{k,j}^{(j)} + \lambda_{k,j}}\right)\left(\frac{\sqrt{\kappa_{k,j}} - 1}{\sqrt{\kappa_{k,j}} + 1}\right)^j,$$

and from [23, eq. (34)] and Assumption 1.1 one finds $\gamma_{k,j+1} \leq \|H_k\| \leq H_{\max}$. Combining these bounds yields the desired conclusion. ∎

We now prove upper bounds on the total number of inner iterations (over $j \in \mathbb{N}$) that are performed during any outer iteration of I-TRACE that corresponds to an iterate that is not ϵ-stationary. For one thing, these bounds serve as upper bounds on the number of Hessian-vector products required during such outer iterations of I-TRACE. They are also part of upper bounds that we prove for the number of function evaluations during each such outer iteration of I-TRACE. The first bound that we prove corresponds to the number of iterations that can be performed until (2.4a) holds, whereas the second bound corresponds—assuming (2.4c) holds—to the number of iterations that can be performed until (2.4b) holds. (As has already been seen in the proof of Lemma 3.2, the condition in (2.4a) is always satisfiable if enough inner iterations are performed, whereas satisfaction of (2.4b)–(2.4c) is not always guaranteed. That said, the algorithm considers (2.4b)–(2.4c) as termination criteria since satisfaction of these inequalities might allow the algorithm to proceed after fewer inner iterations than would be required for (2.4a).)

To state and prove the aforementioned desired bounds, we define two sets. Specifically, for arbitrary $(\bar{\kappa}, \bar{\lambda}) \in \mathbb{R}_{>0} \times \mathbb{R}_{>0}$, let us define the sets of index pairs

$$\mathcal{I}_1(\bar{\kappa}, \bar{\lambda}) := \{(k,j) \in \mathbb{N} \times \mathbb{N} : (k,j) \text{ is generated, } \kappa_{k,j} \leq \bar{\kappa}, \text{ and } \omega_{k,j}^{(j)} + \lambda_{k,j} \leq \bar{\lambda}\}$$
$$\text{and } \mathcal{I}_2(\bar{\kappa}) := \{(k,j) \in \mathbb{N} \times \mathbb{N} : (k,j) \text{ is generated and } \kappa_{k,j} \leq \bar{\kappa}\}.$$

The next lemma shows, at any iterate that is not ϵ-stationary, that if there exists a pair $(\bar{\kappa}, \bar{\lambda})$ such that $(k,j) \in \mathcal{I}_1(\bar{\kappa}, \bar{\lambda})$ for sufficiently large j, then TLTR and the **loop** of I-TRACE terminate before or at inner iteration number j. It also shows the same conclusion under similar conditions when $(k,j) \in \mathcal{I}_2(\bar{\kappa})$ and (2.4c) holds. As shown after the lemma, a consequence of this result is that under nice circumstances, including well-conditioning of the (explicitly or implicitly) regularized reduced-space Hessian, the number of iterations performed by TLTR plus the number of iterations of the **loop** in I-TRACE is $\mathcal{O}(\log(\epsilon^{-1}))$. Otherwise, this sum is at most $\mathcal{O}(n)$.

LEMMA 3.18. *For arbitrary $\epsilon \in (0,1)$ and $k \in \mathbb{N}$ such that $\|g_k\| > \epsilon$, consider the following possible scenarios:*

(i) *There exists $(\bar{\kappa}, \bar{\lambda}) \in \mathbb{R}_{>0} \times \mathbb{R}_{>0}$ such that with*

$$\mathcal{J}_1(\bar{\kappa}, \bar{\lambda}) := \min\left\{n-1, \left\lceil \log\left(\frac{2H_{\max}\bar{\kappa}\bar{\lambda}}{\xi_1\epsilon}\right) \Big/ \log\left(\frac{\sqrt{\bar{\kappa}}+1}{\sqrt{\bar{\kappa}}-1}\right)\right\rceil\right\},$$

one finds that if (k,j) with $j = \mathcal{J}_1(\bar{\kappa}, \bar{\lambda})$ is generated, then $(k,j) \in \mathcal{I}_1(\bar{\kappa}, \bar{\lambda})$.

(ii) *There exists $\bar{\kappa} \in \mathbb{R}_{>0}$ such that with*

$$\mathcal{J}_2(\bar{\kappa}) := \min\left\{n-1, \left\lceil \log\left(\frac{2H_{\max}\bar{\kappa}\xi_3}{\xi_2\epsilon}\right) \Big/ \log\left(\frac{\sqrt{\bar{\kappa}}+1}{\sqrt{\bar{\kappa}}-1}\right)\right\rceil\right\},$$

one finds that if (k,j) with $j = \mathcal{J}_2(\bar{\kappa})$ is generated, then $(k,j) \in \mathcal{I}_2(\bar{\kappa})$ and $1 \le \xi_3 \min\{1, \|t_{k,j}\|\}(\omega_{k,j}^{(j)} + \lambda_{k,j})$.

If scenario (i) (*resp.,* (ii)) *occurs, then each of* TLTR *and the* ***loop*** *of* I-TRACE *terminates before or at inner iteration $j = \mathcal{J}_1(\bar{\kappa}, \bar{\lambda})$ (resp., $j = \mathcal{J}_2(\bar{\kappa})$).*

Proof. Consider arbitrary $k \in \mathbb{N}$ with $\|g_k\| > \epsilon$. That each generated (k,j) has $j \le n-1$ follows from Lemma 3.11. Hence, all that remains is to prove that under the conditions of (i), each generated (k,j) has $j \le \mathcal{J}_1(\bar{\kappa}, \bar{\lambda})$, and that under the conditions of (ii), each generated (k,j) has $j \le \mathcal{J}_2(\bar{\kappa})$.

First, suppose the conditions of (i) hold in the sense that (i) either each generated (k,j) has $j < \mathcal{J}_1(\bar{\kappa}, \bar{\lambda})$ or (ii) both of the following occur: $(k, \mathcal{J}_1(\bar{\kappa}, \bar{\lambda}))$ is generated and $(k, \mathcal{J}_1(\bar{\kappa}, \bar{\lambda})) \in \mathcal{I}_1(\bar{\kappa}, \bar{\lambda})$. Observe that $(\sqrt{\kappa}-1)/(\sqrt{\kappa}+1)$ is a monotonically increasing function of $\kappa \in \mathbb{R}_{>0}$, so by Lemma 3.17 one finds that, for all generated (k,j), one has

$$\|r_{k,j}\| \le \left(\frac{2\|g_k\|H_{\max}\bar{\kappa}}{\omega_{k,j}^{(j)} + \lambda_{k,j}}\right)\left(\frac{\sqrt{\bar{\kappa}}-1}{\sqrt{\bar{\kappa}}+1}\right)^j. \tag{3.3}$$

Consider the case when $\mathcal{J}_1(\bar{\kappa}, \bar{\lambda}) < n-1$ and $j = \mathcal{J}_1(\bar{\kappa}, \bar{\lambda})$, where one finds that

$$\begin{aligned}
& j \ge \log\left(\frac{2H_{\max}\bar{\kappa}\bar{\lambda}}{\xi_1\epsilon}\right) \Big/ \log\left(\frac{\sqrt{\bar{\kappa}}+1}{\sqrt{\bar{\kappa}}-1}\right) \\
\iff \quad & j \log\left(\frac{\sqrt{\bar{\kappa}}+1}{\sqrt{\bar{\kappa}}-1}\right) \ge \log\left(\frac{2H_{\max}\bar{\kappa}\bar{\lambda}}{\xi_1\epsilon}\right) \\
\iff \quad & j \log\left(\frac{\sqrt{\bar{\kappa}}-1}{\sqrt{\bar{\kappa}}+1}\right) \le \log\left(\frac{\xi_1\epsilon}{2H_{\max}\bar{\kappa}\bar{\lambda}}\right) \\
\iff \quad & \left(\frac{\sqrt{\bar{\kappa}}-1}{\sqrt{\bar{\kappa}}+1}\right)^j \le \frac{\xi_1\epsilon}{2H_{\max}\bar{\kappa}\bar{\lambda}}.
\end{aligned} \tag{3.4}$$

Now observe that, by (2.1a), the Cauchy–Schwarz inequality, and the fact that the 2-norm of a real symmetric matrix is its largest eigenvalue, one finds that

$$\begin{aligned}\|g_k\|^2 = \|\gamma_{k,0}e_1\|^2 &= \|(T_{k,j}+\lambda_{k,j}I)t_{k,j}\|^2 \\ &\le \|T_{k,j}+\lambda_{k,j}I\|^2\|t_{k,j}\|^2 = (\omega_{k,j}^{(j)}+\lambda_{k,j})^2\|t_{k,j}\|^2.\end{aligned} \tag{3.5}$$

Hence, under the conditions of (i), one finds that (3.4) implies

$$\begin{aligned}\left(\frac{2\|g_k\|H_{\max}\bar{\kappa}}{\omega_{k,j}^{(j)}+\lambda_{k,j}}\right)\left(\frac{\sqrt{\bar{\kappa}}-1}{\sqrt{\bar{\kappa}}+1}\right)^j &\le \left(\frac{\|g_k\|}{\omega_{k,j}^{(j)}+\lambda_{k,j}}\right)\left(\frac{\xi_1\epsilon}{\bar{\lambda}}\right) \\ &\le \xi_1\left(\frac{\|g_k\|^2}{(\omega_{k,j}^{(j)}+\lambda_{k,j})^2}\right)\left(\frac{\omega_{k,j}^{(j)}+\lambda_{k,j}}{\bar{\lambda}}\right) \le \xi_1\|t_{k,j}\|^2.\end{aligned}$$

Along with (3.3), this bound shows that such a j is sufficiently large such that (2.4a) holds. Therefore, by the construction of I-TRACE, the desired conclusion follows.

Now suppose the conditions of (ii) hold in the sense that either (i) each generated (k,j) has $j < \mathcal{J}_2(\bar{\kappa})$ or (ii) all of the following occur: $(k,\mathcal{J}_2(\bar{\kappa}))$ is generated, $(k,\mathcal{J}_2(\bar{\kappa})) \in \mathcal{I}_2(\bar{\kappa})$, and (since $\omega_{k,j}^{(j)}+\lambda_{k,j} = \|T_{k,j}+\lambda_{k,j}\|$) with $j = \mathcal{J}_2(\bar{\kappa})$ the inequality in (2.4c) holds. A proof similar to that in the previous paragraph applies here as well. First, with $\mathcal{J}_2(\bar{\kappa})$ in place of $\mathcal{J}_1(\bar{\kappa},\bar{\lambda})$, one finds in the present setting that (3.4) becomes

$$\left(\frac{\sqrt{\bar{\kappa}}-1}{\sqrt{\bar{\kappa}}+1}\right)^j \le \frac{\xi_2\epsilon}{2H_{\max}\bar{\kappa}\xi_3}. \tag{3.6}$$

Now, under the conditions of (ii), one finds that (3.6) implies

$$\left(\frac{2\|g_k\|H_{\max}\bar{\kappa}}{\omega_{k,j}^{(j)}+\lambda_{k,j}}\right)\left(\frac{\sqrt{\bar{\kappa}}-1}{\sqrt{\bar{\kappa}}+1}\right)^j \le \frac{\xi_2\epsilon\|g_k\|}{\xi_3(\omega_{k,j}^{(j)}+\lambda_{k,j})} \le \xi_2\min\{1,\|t_{k,j}\|\}\|g_k\|.$$

Along with (3.3) (which applies here as well) and $\|g_k\| = \gamma_{k,0}$, this bound shows that such a j is sufficiently large such that (2.4b) and (2.4c) hold. Therefore, by the construction of I-TRACE, the desired conclusion follows. ∎

One finds in the proof of Lemma 3.18 the reason why the algorithm pairs conditions (2.3b) and (2.3c) or, equivalently, (2.4b) and (2.4c). Using the bound (3.3) from Lemma 3.17 and the fact that for sufficiently large j one obtains (see (3.4)) that

$$\left(\frac{\sqrt{\bar{\kappa}}-1}{\sqrt{\bar{\kappa}}+1}\right)^j = \mathcal{O}(\epsilon),$$

it follows under nice subproblem conditions that for sufficiently large j, one obtains

$$\|r_{k,j}\| = \mathcal{O}\left(\frac{\|g_k\|\epsilon}{\omega_{k,j}^{(j)}+\lambda_{k,j}}\right).$$

Hence, an upper bound for $\mu_{k,j} = \|r_{k,j}\|$ (call it `rhs`) that could be used in a residual condition in order to prove a good worst-case complexity bound for Hessian-vector products is one for which it can be shown that for sufficiently large j, one finds

$$\frac{\|g_k\|\epsilon}{\omega_{k,j}^{(j)}+\lambda_{k,j}} = \mathcal{O}(\texttt{rhs}). \tag{3.7}$$

If $\omega_{k,j}^{(j)} + \lambda_{k,j} \leq \bar{\lambda}$, then one can show with (3.5) that `rhs` on the order of $\|t_{k,j}\|^2$ is sufficient; this is the conclusion of part (i) of the lemma. However, such a residual condition might be stronger than needed, which can be seen by the fact that the proof under the conditions of (i) uses the facts that $\epsilon \leq \|g_k\|$ and the residual condition is based on the larger quantity, namely, $\|g_k\|$. Hence, it is reasonable to wonder whether one can instead impose a residual condition on the order of $\min\{1, \|t_{k,j}\|\}\|g_k\|$, which, as we have already seen in Lemma 3.14 and Theorem 3.15, can lead to desirable complexity properties in terms of iterations and gradient evaluations. The conclusion is that such a residual condition can be employed but only if $\omega_{k,j}^{(j)} + \lambda_{k,j}$ (namely, the denominator in (3.7)) is not too *small* relative to other quantities. Hence, the algorithm pairs the residual condition (2.3b) (resp., (2.4b)) with (2.3c) (resp., (2.4c)), the latter of which ensures that $\omega_{k,j}^{(j)} + \lambda_{k,j}$ is not too small relative to the appropriate quantities, all of which are computable (at modest cost) within the algorithm.

We can now prove a worst-case complexity bound for Hessian-vector products.

THEOREM 3.19. *For arbitrary $\epsilon \in (0,1)$, define the index set $\mathcal{K}(\epsilon)$ and positive integer $K(\epsilon)$ as in Theorem* 3.15. *If there exists uniform $(\bar{\kappa}, \bar{\lambda}) \in \mathbb{R}_{>0} \times \mathbb{R}_{>0}$ such that the conditions in* (i) *and/or* (ii) *of Lemma* 3.18 *hold for all $k \in \mathcal{K}(\epsilon)$, then the total number of Hessian-vector products performed by i-trace* (*and its subroutines*) *at iterates that are not ϵ-stationary is at most*

$$K_H(\epsilon) := K(\epsilon) \cdot \min\{\mathcal{J}_1(\bar{\kappa}, \bar{\lambda}), \mathcal{J}_2(\bar{\kappa})\} = \mathcal{O}(\epsilon^{-3/2} \cdot \min\{n, \log(\epsilon^{-1})\}).$$

Otherwise, if such a $(\bar{\kappa}, \bar{\lambda})$ does not exist, then the number of products is $\mathcal{O}(\epsilon^{-3/2} \cdot n)$.

Proof. The result follows by Theorem 3.15 and Lemmas 3.11 and 3.18. ∎

All that remains for our worst-case analysis is to account for function evaluations that occur through line 2 in FDS. Beyond the results that we have proved already, accounting for function evaluations requires proving an upper bound on the number of iterations that can be performed within FDS. As is proved in the previous subsection, one finds for all generated $(k,j) \in \mathbb{N} \times \mathbb{N}$ that $|\mathcal{A}_{k,j}| = 1$ and $|\mathcal{E}_{k,j}| \leq 1$ (recall Lemma 3.8); hence, what is needed for our purposes here is an upper bound on $|\mathcal{C}_{k,j}|$. A uniform bound over all generated $(k,j) \in \mathbb{N} \times \mathbb{N}$ is proved in the next lemma.

LEMMA 3.20. *For all generated $(k,j) \in \mathbb{N} \times \mathbb{N}$, one finds that*

$$|\mathcal{C}_{k,j}| \leq 1 + \left\lfloor \frac{\log\left(\frac{\sigma_{\max}}{\underline{\sigma}}\right)}{\log\left(\min\left\{\gamma_\lambda, \frac{1}{\gamma_C}\right\}\right)} \right\rfloor =: K_{\mathcal{C}}.$$

Proof. Consider arbitrary generated $(k,j) \in \mathbb{N} \times \mathbb{N}$ such that FDS is called. If $|\mathcal{C}_{k,j}| = 0$, then the desired conclusion follows trivially. Hence, we may proceed under the assumption that $|\mathcal{C}_{k,j}| \geq 1$. It follows by Lemma 3.10 that $|\mathcal{C}_{k,j}| < \infty$. One may also conclude by Lemma 3.7 that $|\mathcal{C}_{k,j}| \geq 1$ means that $\mathcal{C}_{k,j}$ consists of a set of consecutive positive integers. Overall, we may proceed knowing that $\mathcal{C}_{k,j} = \{\underline{l}, \ldots, \bar{l}\}$ for some $(\underline{l}, \bar{l}) \in \mathbb{N} \times \mathbb{N}$. Since it follows by this definition of $\bar{l}$ and Lemma 3.7 that $(\bar{l}+1) \in \mathcal{A}_{k,j}$, it follows with Lemma 3.13 that $\frac{\lambda_{k,j,\bar{l}+1}}{\|t_{k,j,\bar{l}+1}\|} \leq \sigma_{k,j,\bar{l}+1} \leq \sigma_{\max}$. On the other hand, by applying Lemma 3.12 iteratively, one finds that

$$\frac{\lambda_{k,j,\bar{l}+1}}{\|t_{k,j,\bar{l}+1}\|} \geq \underline{\sigma}\left(\min\left\{\gamma_\lambda, \frac{1}{\gamma_C}\right\}\right)^{\bar{l}-\underline{l}}.$$

Combining these upper and lower bounds shows that

$$(\bar{l}-\underline{l})\log\left(\min\left\{\gamma_\lambda,\frac{1}{\gamma_C}\right\}\right)\leq\log\left(\frac{\sigma_{\max}}{\underline{\sigma}}\right)\iff\bar{l}-\underline{l}\leq\frac{\log\left(\frac{\sigma_{\max}}{\underline{\sigma}}\right)}{\log\left(\min\left\{\gamma_\lambda,\frac{1}{\gamma_C}\right\}\right)}.$$

Hence, the desired uniform bound holds since $|\mathcal{C}_{k,j}|=\bar{l}-\underline{l}+1$. ∎

Since, with the previous lemma, there exists a uniform upper bound—independent of the norm of the gradient of the objective—on the number of function evaluations that occur within any inner iteration of I-TRACE, it follows that the worst-case number of function evaluations performed by I-TRACE is of the same order as the number of Hessian-vector products. This is formalized in the following theorem.

THEOREM 3.21. *For arbitrary $\epsilon\in(0,1)$, define the index set $\mathcal{K}(\epsilon)$ and positive integer $K(\epsilon)$ as in Theorem* 3.15. *If there exists uniform $(\bar{\kappa},\bar{\lambda})\in\mathbb{R}_{>0}\times\mathbb{R}_{>0}$ such that the conditions in* (i) *and/or* (ii) *of Lemma* 3.18 *hold for all $k\in\mathcal{K}(\epsilon)$, then the total number of function evaluations performed by* I-TRACE *(and its subroutines) at iterates that are not ϵ-stationary is at most*

$$K_f(\epsilon):=K_{\mathcal{C}}\cdot K(\epsilon)\cdot\min\{\mathcal{J}_1(\bar{\kappa},\bar{\lambda}),\mathcal{J}_2(\bar{\kappa})\}=\mathcal{O}(\epsilon^{-3/2}\cdot\min\{n,\log(\epsilon^{-1})\}).$$

Otherwise, if such a $(\bar{\kappa},\bar{\lambda})$ does not exist, then the number of evaluations is $\mathcal{O}(\epsilon^{-3/2}\cdot n)$.

Proof. The result follows by Theorem 3.15 and Lemmas 3.11, 3.18, and 3.20. ∎

3.3. Local convergence. I-TRACE can attain the same local convergence rate to a strict local minimizer that is attained by TRACE. This property of I-TRACE follows using well-known results from analyses of inexact Newton methods; nonetheless, it is important to state the results for the sake of completeness.

Our presentation here borrows from that in [14, section 3.4]. We consider the local convergence rate attainable by I-TRACE under the following assumption.

Assumption 3.2. With respect to an infinite index set $\mathcal{S}\subseteq\mathbb{N}$, the iterate subsequence $\{x_k\}_{k\in\mathcal{S}}$ converges to $x_*\in\mathbb{R}^n$ at which $H(x_*)\succ 0$. In addition, there exists a nonempty neighborhood of x_* over which the Hessian function H is locally Lipschitz continuous with Lipschitz constant $H_{\mathrm{Loc}}\in\mathbb{R}_{>0}$.

The following lemma captures a property of TRACE inherited by I-TRACE.

LEMMA 3.22. *Under Assumption* 3.2, *the entire sequence $\{x_k\}$ converges to x_*.*

Proof. As previously mentioned at the beginning of subsection 3.2, the analysis there shows that $\{\|g_k\|\}\to 0$, which under Assumption 3.2 implies $g(x_*)=0$. As in the context of [14, Lemma 3.30], the remainder of the proof follows similarly to that of [11, Theorem 6.5.2]. ∎

Our next lemma is similar to [14, Lemma 3.31] insofar as it shows that, eventually, all computed steps are (potentially inexact) Newton steps that are accepted by the algorithm. Our proof follows closely that of [14, Lemma 3.31] but with modifications to account for the potential inexactness of the computed subproblem solutions.

LEMMA 3.23. *There exists $k_*\in\mathbb{N}$ such that, for all $k\in\mathbb{N}$ with $k\geq k_*$, line* 8 *of i-trace is reached with $\lambda_{k,j}=0$ and $|\mathcal{C}_{k,j}|=|\mathcal{E}_{k,j}|=0$.*

Proof. By Lemma 3.22, the iterate sequence $\{x_k\}$ converges to x_*, at which it follows under Assumption 3.2 that $H_*:=H(x_*)\succ 0$. Let the smallest and largest eigenvalues of H_* be denoted by $\omega_{\min}$ and $\omega_{\max}$, respectively. By continuity of H, it follows

that the eigenvalues of H_k are contained within the positive interval $[\frac{1}{2}\omega_{\min}, 2\omega_{\max}]$ for all sufficiently large $k \in \mathbb{N}$. Consider an arbitrary such k and consider arbitrary $j \in \mathbb{N}$ such that the index pair (k,j) is generated and FDS is called. Due to the aforementioned property of the eigenvalues of H_k, it follows (see [23, section 3.2]) that the eigenvalues of $T_{k,j}$ are contained in $[\frac{1}{2}\omega_{\min}, 2\omega_{\max}]$ as well. Consider now arbitrary generated $l \in \mathbb{N}$. Either $\|t_{k,j,l}\| = \delta_{k,j,l}$ or $t_{k,j,l} = -T_{k,j}^{-1}(\gamma_{k,0}e_1)$; either way,

$$\|t_{k,j,l}\| \leq \|T_{k,j}^{-1}(\gamma_{k,0}e_1)\| \leq \|T_{k,j}^{-1}\|\|g_k\| \implies \|g_k\| \geq \|t_{k,j,l}\|/\|T_{k,j}^{-1}\|. \tag{3.8}$$

By standard trust-region theory pertaining to Cauchy decrease, it now follows that

$$\begin{aligned} f(x_k) - m_k(Q_{k,j}t_{k,j,l}) &\geq \frac{1}{2}\|g_k\| \min\left\{\delta_{k,j,l}, \frac{\|g_k\|}{\|T_{k,j}\|}\right\} \\ &\geq \frac{1}{2}\left(\frac{\|t_{k,j,l}\|}{\|T_{k,j}^{-1}\|}\right) \min\left\{\|t_{k,j,l}\|, \frac{\|t_{k,j,l}\|}{\|T_{k,j}\|\|T_{k,j}^{-1}\|}\right\} \\ &\geq \frac{1}{2}\left(\frac{\|t_{k,j,l}\|^2}{\|T_{k,j}\|\|T_{k,j}^{-1}\|^2}\right) \geq \frac{1}{16}\omega_{\max}^{-1}\omega_{\min}^2\|t_{k,j,l}\|^2 =: \eta_*\|t_{k,j,l}\|^2. \end{aligned}$$

One also finds from (3.8), the fact that $\{\|g_k\|\} \to 0$, and the aforementioned properties of the eigenvalues of $T_{k,j}$ that for any $\varepsilon \in (0,1)$ there exists sufficiently large $k_\varepsilon \in \mathbb{N}$ such that $\|t_{k,j,l}\| \leq \varepsilon$ for all generated (k,j,l) with $k \geq k_\varepsilon$. Combining these facts shows, using an argument similar to the proof of Lemma 3.6, that for sufficiently large $k \in \mathbb{N}$ one finds for any generated (k,j,l) that

$$\begin{aligned} f_k - f(x_k + Q_{k,j}t_{k,j,l}) &\geq f_k - m_k(Q_{k,j}t_{k,j,l}) + m_k(Q_{k,j}t_{k,j,l}) - f(x_k + Q_{k,j}t_{k,j,l}) \\ &\geq \eta_*\|t_{k,j,l}\|^2 - \tfrac{1}{2}H_{\mathrm{Loc}}\|t_{k,j,l}\|^3 \geq \eta\|Q_{k,j}t_{k,j,l}\|^3. \end{aligned}$$

It follows from this fact that, for any such generated (k,j,l), one has $l \in \mathcal{A}_{k,j} \cup \mathcal{E}_{k,j}$.

By the results of the previous paragraph, there exists $\delta_{\min} \in \mathbb{R}_{>0}$ such that $\delta_{k,j,l} \geq \delta_{\min}$ for all generated (k,j,l) with sufficiently large k. In addition, continuity of g and the aforementioned properties of the eigenvalues of $T_{k,j}$ imply that the trial step $t_{k,j,l}$ lies in the interior of the trust region for all generated (k,j,l) with sufficiently large k. Since this means that $\lambda_{k,j,l} = 0$ for all such generated (k,j,l), it follows that, in fact, for all generated (k,j,l) for sufficiently large k one has $l \in \mathcal{A}_{k,j}$. ∎

We now use the standard theory of inexact Newton methods to show that I-TRACE can, e.g., attain the same rate of local convergence as TRACE (see [14, Theorem 3.32]).

THEOREM 3.24. *If, in addition to* (2.4)*, the* ***if*** *condition in line* 6 *of* I-TRACE *requires* $\mu_{k,j} = o(\|g_k\|)$*, then* $\{x_k\} \to x_*$ *Q-superlinearly. In particular, if the condition requires* $\mu_{k,j} = \mathcal{O}(\|g_k\|^2)$*, then* $\{x_k\} \to x_*$ *Q-quadratically.*

Proof. With Lemmas 3.22 and 3.23, the conclusion follows using the standard theory of inexact Newton methods; see [16, Theorem 3.3]. ∎

4. Numerical results. In this section, we provide the results of numerical experiments of a prototype implementation of I-TRACE, as well as implementations of TRACE [14], ARC [8, 9], and NEWTON-CG [13] for the sake of comparison. The purposes of presenting these experimental results are twofold. First, we show that, by allowing inexact subproblem solutions, I-TRACE offers computational flexibility beyond that offered by TRACE. Second, we show that, in terms of key performance measures, I-TRACE performs at least as well as ARC, which is a state-of-the-art second-order

method that offers optimal worst-case iteration, function-evaluation and derivative-evaluation complexities to ϵ-stationarity, and at least as well as NEWTON-CG, which offers state-of-the-art complexity to first- and second-order stationarity (although our implementation ignores the minimum-eigenvalue oracle as described in [13] and only focuses on attaining (first-order) ϵ-stationarity). For these experiments, all of the algorithms were implemented in a single software package in MATLAB. All experiments were run using the `polyps` cluster at Lehigh University's COR@L Laboratory. Each job was run with a wall-clock-time limit of 90 minutes and a memory limit of 8 GB.

4.1. Implementation details. The implementations of I-TRACE and TRACE share many commonalities. For a fair comparison, both implementations involve the auxiliary sequence $\{\Delta_k\}$, the values of which are set and used as in [14, Algorithm 1]. As explained in the last paragraph of section 2, the theoretical guarantees that have been proved in this paper are maintained with the inclusion of this auxiliary sequence and, in fact, allow one to prove guarantees under weaker assumptions. For our experiments, the common parameters for I-TRACE and TRACE were set as $\eta = 10^{-4}$, $\underline{\sigma} = 0.01$, $\overline{\sigma} = 100$, $\gamma_C = 0.5$, $\gamma_E = 1.1$, $\gamma_\lambda = 2$, $\delta_0 = 1$, $\sigma_0 = 1$, and $\Delta_0 = 100$. Specifically for I-TRACE, we ran experiments for $(\xi_1, \xi_2) \in \{(0.1, 0.01), (1, 0.1), (9, 0.9)\}$ and $\xi_3 = 10^6$. (This choice of ξ_3 is intended to ensure that (2.4c) is often satisfied so that the inexactness condition essentially requires that either (2.4a) or (2.4b) holds, which, in turn, means that performance depends on the setting for (ξ_1, ξ_2).) For the implementation of TRACE, all trust-region subproblems are solved using an implementation of the Moré–Sorensen approach [29]. For the implementation of I-TRACE, each $\mathcal{R}_{k,j}$ subproblem is solved by solving a tridiagonal system, each $\mathcal{S}_{k,j}(\delta)$ subproblem is solved using the aforementioned implementation of the Moré–Sorensen approach, and each instance of the subproblem in line 16 of FDS is solved using an implementation of [8, Algorithm 6.1], where, as described in [14], the algorithm is terminated as soon as the ratio $\frac{\bar{\lambda}_{k,j,l+1}}{\|\bar{t}_{k,j,l+1}\|}$ lies in the interval $[\underline{\sigma}, \overline{\sigma}]$.

For the implementation of ARC, the parameters were set as $\eta_1 = 10^{-4}$, $\eta_2 = 0.9$, and $\sigma_0 = 1$. In ARC, $\{\sigma_k\}$ is the sequence of cubic regularization values that is updated dynamically by the algorithm. In our implementation, this sequence is updated as for the experiments in [8], namely, $\sigma_{k+1} \leftarrow \max\{\min\{\sigma_k, \|g_k\|, 10^{-16}\}\}$ if k is a very successful iteration; $\sigma_{k+1} \leftarrow \sigma_k$ if k is successful (but not very successful) iteration; and $\sigma_{k+1} \leftarrow 2\sigma_k$ if k is an unsuccessful iteration. As for I-TRACE, the subproblems are solved using an iterative method that employs the Lanczos approach, where for a termination condition our implementation employs `TC.s` (stated as [8, eq. (3.28)]), which involves the user-defined parameter κ_θ. Note that `TC.s` is the same as (2.4b) with $\kappa_\theta \equiv \xi_2$. Experiments comparable to those for I-TRACE, were run with $\kappa_\theta \in \{0.01, 0.1, 0.9\}$.

For the implementation of NEWTON-CG, we set $\epsilon_g = 10^{-5}$, $\epsilon_H = \sqrt{10^{-5}}$, $\gamma_1 = 0.5$, $\gamma_2 = 2$, $\psi = 0.75$, $\delta_0 = 1$, $\delta_{\max} = 10^8$, $\eta = 0.1$, and `capCG = false`, while for ζ, which controls the residual tolerance for CG, we ran experiments with $\zeta \in \{0.01, 0.1, 0.9\}$. It is important to note that our implementation of NEWTON-CG does not call a minimum eigenvalue oracle (MEO) as described in [13], since for consistency with the other algorithms our implementation focuses only on first-order stationarity. In this manner, in our implementation of NEWTON-CG, if the CG iteration limit (of n) is reached, then the search direction is set as the last CG iterate.

All implemented algorithms respect the same termination condition, namely,

$$\|g_k\| \leq 10^{-5} \max\{1, \|g_0\|\}. \tag{4.1}$$

4.2. Computational flexibility offered by inexactness. Our first set of experiments demonstrates the computational flexibility that I-TRACE allows over TRACE due to the fact that I-TRACE can employ inexact subproblem solutions. For this experiment, we ran I-TRACE with all parameter settings (see the choices of (ξ_1, ξ_2) in the previous subsection, respectively referred to as "setting 1," "setting 2," and "setting 3") and TRACE to solve all of the unconstrained instances in the CUTEst [21] collection (with their original parameter settings). This originally includes 238 problems. To give a sense of the dimensions of these problems, we note that 103 have less than or equal to 10 variables, 32 have between 11 and 100 variables, 13 have between 101 and 1000 variables, and 90 have more than 1000 variables. Defining success as encountering an iterate satisfying (4.1), I-TRACE with setting 1 successfully solved 214 problems, I-TRACE with setting 2 solved 218 problems, I-TRACE with setting 3 solved 219 problems, and TRACE solved only 188 problems due to hitting the time or memory limit much more often than I-TRACE. (For future reference, we note that ARC—run with the three aforementioned values of κ_θ respectively referred to as settings 1, 2, and 3—solved 210 problems with setting 1, 214 problems with setting 2, and 216 problems with setting 3, whereas NEWTON-CG—run with the three aforementioned values of ζ respectively referred to as settings 1, 2, and 3—solved 225 problems with settings 1 and 2 and solved 226 problems with setting 3.) In the experimental results provided in this section, unless otherwise stated, the results are provided for all problems that were successfully solved by at least one algorithm for one of its settings (including I-TRACE, TRACE, ARC, and NEWTON-CG), which includes 227 problems.

To compare the relative performance of I-TRACE and TRACE, we provide in Figure 1 sets of Dolan–Moré performance profiles [17] for function evaluations, gradient evaluations, and Hessian-vector products. (We limit the horizontal axis to $\tau = 20$ so that the differences between the graphs can be seen more clearly.) In the top row of Figure 1, the profiles are constructed to include the results of our entire set of 227 problems. These profiles show that TRACE is less reliable, which is due to the fact that it hit our time or memory limit much more often. In the bottom row of Figure 1, the profiles are constructed to include only those problems for which I-TRACE with all settings and TRACE were all successful, which is a set of 188 problems.

The bottom row of profiles in Figure 1 shows that, despite allowing inexact subproblem solutions, I-TRACE performs comparably to TRACE in terms of function and gradient evaluations, which also means that the algorithms/settings perform comparably in terms of iterations required. In terms of Hessian-vector products, I-TRACE with setting 1 falls a bit behind the other settings, which we contend is due to the algorithm requiring more accurate subproblem solutions. That said, I-TRACE with setting 1 performs better in terms of gradient evaluations. These results demonstrate, as mentioned in section 1, that I-TRACE offers flexibility between derivative evaluations and Hessian-vector products. A user can choose the parameters that are preferable depending on the relative costs of these operations for a given problem.

For completeness, we provide in Figures 2 and 3 similar sets of performance profiles for ARC and NEWTON-CG that show differences due to parameter settings.

4.3. Comparison with state-of-the-art optimal-complexity algorithms. In this section, we compare the performance of I-TRACE, ARC, and NEWTON-CG. We provide in Figures 4, 5, and 6 performance profiles comparing I-TRACE, ARC, and NEWTON-CG with their settings 1, 2, and 3, respectively. While the settings for the algorithms are not directly comparable due to the variations of the methods (e.g., "setting 1" for I-TRACE is not directly comparable to that for ARC or NEWTON-CG),

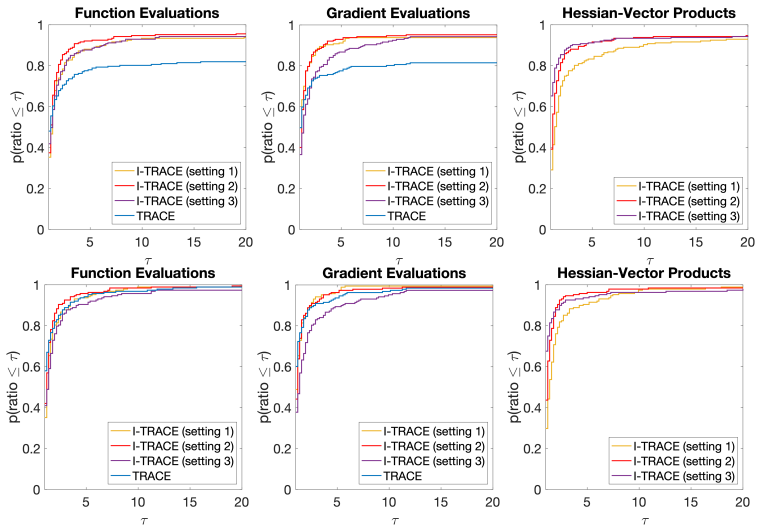

FIG. 1. *Performance profiles for function evaluations, gradient evaluations, and Hessian-vector products for* I-TRACE (*with three parameter settings*) *and* TRACE *when solving* 227 (*top row*) *and* 188 (*bottom row*) *CUTEst problems.* (*trace is not included in the profile for Hessian-vector products since it does not employ Krylov subspace techniques for solving the arising subproblems.*)

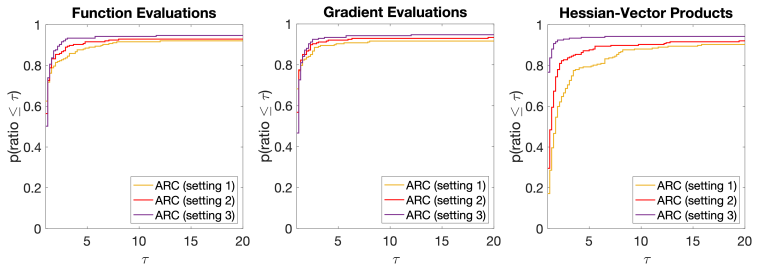

FIG. 2. *Performance profiles for function evaluations, gradient evaluations, and Hessian-vector products for arc* (*with three parameter settings*) *when solving* 227 *CUTEst problems.*

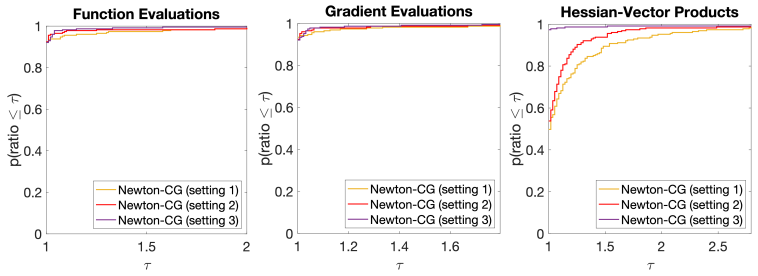

FIG. 3. *Performance profiles for function evaluations, gradient evaluations, and Hessian-vector products for* NEWTON-CG (*with three parameter settings*) *when solving* 227 *CUTEst problems.*

we compare the algorithms in this manner in order to provide, in a straightforward manner, a broad spectrum of comparisons across different inexactness levels. In practice, any user of such algorithms should decide which performance measure is most important for their application, tune each algorithm with respect to that measure, and compare tuned versions of each algorithm for their own purpose.

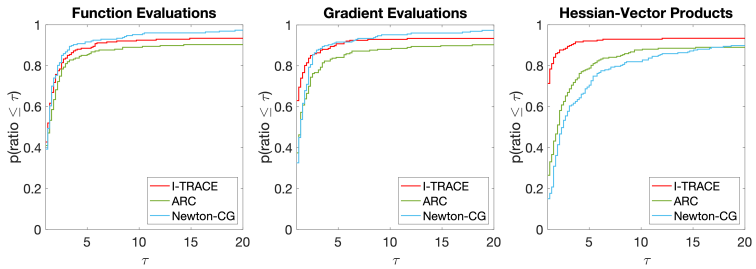

FIG. 4. *Performance profiles for function evaluations, gradient evaluations, and Hessian-vector products for* I-TRACE, ARC, *and* NEWTON-CG (*all with setting* 1) *for* 227 *CUTEst problems.*

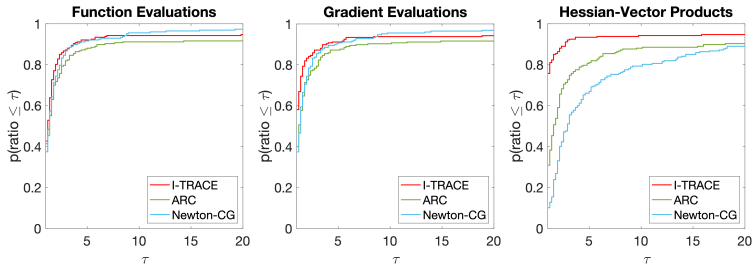

FIG. 5. *Performance profiles for function evaluations, gradient evaluations, and Hessian-vector products for* I-TRACE, ARC, *and* NEWTON-CG (*all with setting* 2) *for* 227 *CUTEst problems.*

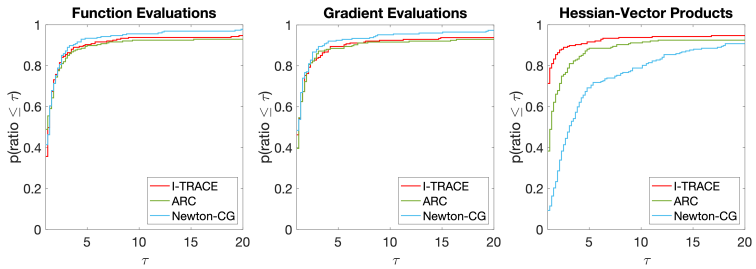

FIG. 6. *Performance profiles for function evaluations, gradient evaluations, and Hessian-vector products for* I-TRACE, ARC, *and* NEWTON-CG (*all with setting* 3) *for* 227 *CUTEst problems.*

The profiles in Figures 4, 5, and 6 show that I-TRACE performs at least as well as ARC and NEWTON-CG across a range of parameter settings and a broad spectrum of problems. In all comparisons, the differences in terms of function and gradient evaluations are relatively minor and are essentially due to the differences in the numbers of problems that each solves successfully. On the other hand, the differences in terms of Hessian-vector products is more noticeable. NEWTON-CG, in particular, does not perform as well as the other algorithms in terms of this measure. We attribute this to the fact that, in our experiments, the CG residual condition in NEWTON-CG (see Algorithm 3.1 in [13]) ends up being difficult to satisfy within relatively few CG iterations, regardless of the value of ζ. This is perhaps due to the CG residual condition in NEWTON-CG depending on the value of ϵ_H.

5. Conclusion. We presented, analyzed, and tested a new algorithm for solving smooth unconstrained optimization problems. The algorithm is an extension of

TRACE [14], specifically one that allows the use of inexact subproblem solutions that are computed using an iterative linear algebra technique (the Lanczos algorithm, a Krylov subspace method). The algorithm, referred to as I-TRACE, maintains the worst-case iteration complexity guarantees (to ϵ-stationarity, as defined in (1.2)) and local convergence rate guarantees of TRACE but offers worst-case guarantees in terms of Hessian-vector products that can be significantly better than those offered by TRACE. Numerical experiments show that I-TRACE can offer better computational trade-offs than TRACE and show that I-TRACE is competitive with a state-of-the-art second-order method with optimal complexity guarantees to ϵ-stationarity.

The worst-case complexity guarantee to ϵ-stationarity offered by I-TRACE is different from that proved for some other algorithms in the literature. To understand why, one should observe that I-TRACE has been designed *with first-order guarantees in mind* to maintain the $\mathcal{O}(\epsilon^{-3/2})$ iteration complexity of TRACE but to improve upon the computational complexity of TRACE by (potentially) requiring fewer than n "inner" iterations per "outer" iteration. This leads to I-TRACE's proved (under certain conditions) Hessian-vector-product complexity of $\mathcal{O}(\epsilon^{-3/2} \cdot \min\{n, \log(\epsilon^{-1})\})$ rather than the more pessimistic $\mathcal{O}(\epsilon^{-3/2} \cdot n)$ complexity that follows when exact subproblem solutions are always required. Other algorithms offer a different complexity bound, such as the $\mathcal{O}(\epsilon^{-3/2} \cdot \log(\epsilon^{-1}) \cdot \min\{n, \epsilon^{-1/4}\})$ bound offered by algorithms such as those in [5, 13, 32]. An explanation for the differences is that, in certain circumstances, these other algorithms either employ so-called negative-curvature steps or call minimum-eigenvalue oracles for computing search directions, and the corresponding subroutines affect the computational complexity in alternative ways. On the other hand, these algorithms also offer complexity guarantees to approximate second-order stationarity, which is not something that is offered for I-TRACE. One could equip I-TRACE with a procedure that would, e.g., call a minimum-eigenvalue oracle when $\|g_k\| \leq \epsilon$ to either (i) verify that the current iterate is approximately second-order stationary or (ii) compute a negative-curvature step otherwise. We have not considered such an extension in this paper since (approximate) second-order guarantees are not regularly required in practical implementations, and because we believe it is worthwhile to contribute to the literature with this work that shows the theoretical and practical advantages of I-TRACE that are motivated purely by first-order guarantees.

REFERENCES

[1] S. Adachi, S. Iwata, Y. Nakatsukasa, and A. Takeda, *Solving the trust-region subproblem by a generalized eigenvalue problem*, SIAM J. Optim., 27 (2017), pp. 269–291, https://doi.org/10.1137/16M1058200.

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

[3] E. G. Birgin and J. M. Martínez, *The use of quadratic regularization with a cubic descent condition for unconstrained optimization*, SIAM J. Optim., 27 (2017), pp. 1049–1074, https://doi.org/10.1137/16M110280X.

[4] Y. Carmon and J. C. Duchi, *Analysis of Krylov subspace solutions of regularized nonconvex quadratic problems*, in Proceedings of the 32nd International Conference on Neural Information Processing Systems, 2018, pp. 10728–10738.

[5] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford, *Accelerated methods for nonconvex optimization*, SIAM J. Optim., 28 (2018), pp. 1751–1772, https://doi.org/10.1137/17M1114296.

[6] C. Cartis, N. I. M. Gould, and Ph. L. Toint, *On the complexity of steepest descent, Newton's and regularized Newton's methods for nonconvex unconstrained optimization problems*, SIAM J. Optim., 20 (2010), pp. 2833–2852, https://doi.org/10.1137/090774100.

[7] C. Cartis, N. I. M. Gould, and Ph. L. Toint, *Evaluation Complexity of Algorithms for Nonconvex Optimization: Theory, Computation and Perspectives*, MOS-SIAM Ser. Optim. 30, SIAM, Philadelphia, 2022, https://doi.org/10.1137/1.9781611976991.

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

[9] C. Cartis, N. I. M. Gould, and Ph. L. Toint, *Adaptive cubic regularisation methods for unconstrained optimization. Part* II*: Worst-case function- and derivative-evaluation complexity*, Math. Program., 130 (2011), pp. 295–319.

[10] A. R. Conn, N. I. M. Gould, and Ph. L. Toint, *A primal-dual algorithm for minimizing a non-convex function subject to bound and linear equality constraints*, in Nonlinear Optimization and Related Topics (Erice, 1998), Appl. Optim. 36, Kluwer Academic Publishers, Dordrecht, 2000, pp. 15–49.

[11] A. R. Conn, N. I. M. Gould, and Ph. L. Toint, *Trust-Region Methods*, MOS-SIAM Ser. Optim. 1, SIAM, Philadelphia, 2000, https://doi.org/10.1137/1.9780898719857.

[12] F. E. Curtis, Z. Lubberts, and D. P. Robinson, *Concise complexity analyses for trust region methods*, Optim. Lett., 12 (2018), pp. 1713–1724, //doi.org/10.1007/s11590-018-1286-2.

[13] F. E. Curtis, D. P. Robinson, C. W. Royer, and S. J. Wright, *Trust-region Newton-CG with strong second-order complexity guarantees for nonconvex optimization*, SIAM J. Optim., 31 (2021), pp. 518–544, //https://doi.org/10.1137/19M130563X.

[14] F. E. Curtis, D. P. Robinson, and M. Samadi, *A trust region algorithm with a worst-case iteration complexity of* $\mathcal{O}(\epsilon^{-3/2})$ *for nonconvex optimization*, Math. Program., 162 (2017), pp. 1–32, https://doi.org/10.1007/s10107-016-1026-2.

[15] F. E. Curtis, D. P. Robinson, and M. Samadi, *An inexact regularized Newton framework with a worst-case iteration complexity of* $\mathcal{O}(\epsilon^{-3/2})$ *for nonconvex optimization*, IMA J. Numer. Anal., 39 (2018), pp. 1296–1327, https://doi.org/doi:10.1093/imanum/dry022.

[16] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408, https://doi.org/10.1137/0719025.

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

[18] J.-P. Dussault, *ARCq: A new adaptive regularization by cubics*, Optim. Methods Softw., 33 (2018), pp. 322–335, https://doi.org/10.1080/10556788.2017.1322080.

[19] J.-P. Dussault and D. Orban, *Scalable Adaptive Cubic Regularization Methods*, Technical report G-2015-109, GERAD, 2015.

[20] N. I. M. Gould, S. Lucidi, M. Roma, and Ph. L. Toint, *Solving the trust-region subproblem using the Lanczos method*, SIAM J. Optim., 9 (1999), pp. 504–525, https://doi.org/10.1137/S1052623497322735.

[21] N. I. M. Gould, D. Orban, and Ph. L. Toint, *CUTEst: A Constrained and Unconstrained Testing Environment with Safe Threads*, Technical report, Rutherford Appleton Laboratory, Chilton, England, 2013, https://doi.org/10.1007/s10589-014-9687-3.

[22] N. I. M. Gould, M. Porcelli, and Ph. L. Toint, *Updating the regularization parameter in the adaptive cubic regularization algorithm*, Comput. Optim. Appl., 53 (2012), pp. 1–22.

[23] N. I. M. Gould and V. Simoncini, *Error estimates for iterative algorithms for minimizing regularized quadratic subproblems*, Optim. Methods Softw., 35 (2020), pp. 304–328.

[24] G. N. Grapiglia, J. Yuan, and Y. Yuan, *On the convergence and worst-case complexity of trust-region and regularization methods for unconstrained optimization*, Math. Program., 152 (2015), pp. 491–520.

[25] A. Griewank, *The Modification of Newton's Method for Unconstrained Optimization by Bounding Cubic Terms*, Technical report NA/12, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, 1981.

[26] Z. Jia and F. Wang, *The convergence of the generalized Lanczos trust-region method for the trust-region subproblem*, SIAM J. Optim., 31 (2021), pp. 887–914, https://doi.org/10.1137/19M1279691.

[27] J. Kuczyński and H. Woźniakowski, *Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 1094–1122, https://doi.org/10.1137/0613066.

[28] J. Kuczyński and H. Woźniakowski, *Probabilistic bounds on the extremal eigenvalues and condition number by the Lanczos algorithm*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 672–691, https://doi.org/10.1137/S0895479892230456.

[29] J. J. Moré and D. C. Sorensen, *Computing a trust region step*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 553–572, https://doi.org/10.1137/0904038.

[30] Y. Nesterov and B. T. Polyak, *Cubic regularization of Newton's method and its global performance*, Math. Program., 108 (2006), pp. 117–205.

[31] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer Ser. Oper. Res. Financ. Eng., Springer, New York, 2006.

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

[33] C. W. Royer and S. J. Wright, *Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization*, SIAM J. Optim., 28 (2018), pp. 1448–1477, https://doi.org/10.1137/17M1134329.

[34] T. Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637, https://doi.org/10.1137/0720042.

[35] Ph. L. Toint, *Towards an efficient sparsity exploiting Newton method for minimization*, in Sparse Matrices and Their Uses, I. S. Duff, ed., Academic Press, London, New York, 1981, pp. 57–88.

[36] L.-H. Zhang, C. Shen, and R.-C. Li, *On the generalized Lanczos trust-region method*, SIAM J. Optim., 27 (2017), pp. 2110–2142, https://doi.org/10.1137/16M1095056.