2018

Efficient Trust Region Methods for Nonconvex Optimization

Mohammadreza Samadi
Lehigh University, samadi87@gmail.com

Efficient Trust Region Methods for Nonconvex Optimization

by

Mohammadreza Samadi

A Dissertation
Presented to the Graduate Committee
of Lehigh University
in Candidacy for the Degree of
Doctor of Philosophy
in
Industrial and Systems Engineering

Lehigh University
January 2019

Approved and recommended for acceptance as a dissertation in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Mohammadreza Samadi
Efficient Trust Region Methods for Nonconvex Optimization

Date

Dr. Frank E. Curtis, Dissertation Director, Chair
(Must Sign with Blue Ink)

Accepted Date

Committee Members

Dr. Frank E. Curtis, Committee chair

Dr. Daniel P. Robinson

Dr. Katya Scheinberg

Dr. Martin Takáč

Acknowledgment

This dissertation would have not been possible without wise advice of Dr. Frank E. Curtis. I have been extremely privileged to have Frank as my advisor. He was a wonderful teacher, a wise mentor, a thoughtful friend, and a supportive brother to me. He patiently taught me numerous lessons not only related to this dissertation but also necessary for succeeding in my professional life.

I would also like to express my gratitude to the remaining members of my dissertation committee, Dr. Daniel P. Robinson, Dr. Katya Scheinberg, and Dr. Martin Takáč for their insightful comments in enriching this dissertation.

In addition, I would like to thank my mentors at SAS Institute Inc., Dr. Yan Xu and Dr. Joshua Griffin, who taught me invaluable lessons during my internship.

Studying Ph.D. was so daunting to me that, if it was not because of all the support provided by my wife, Hiva Ghanbari, I would had quit even before starting. Hiva brought the warmth of home with herself into my life. She inspired me to extend beyond my imagination, accompanied me to fly above all the boarders, and empowered me to shatter the barriers that were keeping me away from living the life of my dreams. She provided me with the courage, joy, and passion enabling me to excel in life. She also helped me in along the path by sharing her insight through many discussions on the topics covered in this dissertation and beyond. There is no word rich enough to express the depth of my gratitude to Hiva. Suffice it to say that I feel extremely lucky to have her in my life as my wife and best friend.

I am tremendously grateful to have the best and most supportive family. Hiva was the main source of motivation for me but she was not alone. My father, Ahmad Samadi, taught me curiosity and integrity. My mother, Soghra Samadi, is constantly giving me the love and support necessary to become a compassionate professional. My brother,

Alireza Samadi, is enriching my dreams with patiently listening and eagerly improving my expectations of life. My sister, Toktam Samadi, is a calming force in my life with her unconditional support. My sister, Fatemeh Samadi, fills me with excitement and joy every time I talk to her. My sister, Soraya Samadi and her own family, helped me in completing my Bachelors and Masters degrees by hosting me at her home for six years. Finally, my sister, Masumeh Samadi, accelerated my graduation by constantly asking "when are you going to be done with the school?".

I have been lucky to have the nicest friends, which I take this opportunity to thank them here, in alphabetic order: Suresh Bolusani, Pelin and Sertalp Cay, Choat Inthawongse, Majid Jahani, Xiaolong Kuang, Matt Menickelly, Mohsen Moarefdoost, Ali Mohammad-Nezhad, Golnaz Shahidi, Shu Tu, Wei Xia, and Alireza Yektamaram.

Yes! "It takes a village" to earn a Ph.D. degree.

Contents

List of Tables

List of Figures

Abstract

For decades, a great deal of nonlinear optimization research has focused on modeling and solving convex problems. This has been due to the fact that convex objects typically represent satisfactory estimates of real-world phenomenon, and since convex objects have very nice mathematical properties that makes analyses of them relatively straightforward. However, this focus has been changing. In various important applications, such as large-scale data fitting and learning problems, researchers are starting to turn away from simple, convex models toward more challenging nonconvex models that better represent real-world behaviors and can offer more useful solutions.

To contribute to this new focus on nonconvex optimization models, we discuss and present new techniques for solving nonconvex optimization problems that possess attractive theoretical and practical properties. First, we propose a trust region algorithm that, in the worst case, is able to drive the norm of the gradient of the objective function below a prescribed threshold of $\epsilon \in (0, \infty)$ after at most $\mathcal{O}(\epsilon^{-3/2})$ iterations, function evaluations, and derivative evaluations. This improves upon the $\mathcal{O}(\epsilon^{-2})$ bound known to hold for some other trust region algorithms and matches the $\mathcal{O}(\epsilon^{-3/2})$ bound for the recently proposed Adaptive Regularisation framework using Cubics, also known as the ARC algorithm. Our algorithm, entitled TRACE, follows a trust region framework, but employs modified step acceptance criteria and a novel trust region update mechanism that allow the algorithm to achieve such a worst-case global complexity bound. Importantly, we prove that our algorithm also attains global and fast local convergence guarantees under similar assumptions as for other trust region algorithms. We also prove a worst-case upper bound on the number of iterations the algorithm requires to obtain an approximate second-order stationary point.

The aforementioned algorithm is based on techniques that require an exact subproblem

solution in every iteration. This is a reasonable assumption for small- to medium-scale problems, but is intractable for large-scale optimization. To address this issue, the second project of this thesis involves a proposal of a general *inexact* framework, which contains a wide range of algorithms with optimal complexity bounds, through defining a novel primal-dual subproblem and a set of loose conditions for an inexact solution of it. The proposed framework enjoys the same worst-case iteration complexity bounds for locating approximate first- and second-order stationary points as TRACE. However, it does not require one to solve subproblems exactly. In addition, the framework allows one to use inexact Newton steps whenever possible, a feature which allows the algorithm to use Hessian matrix-free approaches such as the *conjugate gradient* method. This improves the practical performance of the algorithm, as our numerical experiments show.

We close by proposing a globally convergent trust funnel algorithm for equality constrained optimization. The proposed algorithm, under some standard assumptions, is able to find a relative first-order stationary point after at most $\mathcal{O}(\epsilon^{-3/2})$ iterations. This matches the complexity bound of the recently proposed Short-Step ARC algorithm. Our proposed algorithm uses the step decomposition and feasibility control mechanism of a trust funnel algorithm, but incorporates ideas from our TRACE framework in order to achieve good complexity bounds.

Chapter 1

Introduction

1.1 Motivation

For decades, the primary aim for many researchers working on solving nonconvex smooth optimization problems has been to design numerical methods that attain global and fast local convergence guarantees. Indeed, a wide variety of such methods have been proposed, many of which can be characterized as being built upon steepest descent, Newton, or quasi-Newton methodologies, generally falling into the categories of line search and trust region methods. For an extensive background on such ideas, one need only refer to numerous textbooks that have been written on nonlinear optimization theory and algorithms; e.g., see [1, 3, 23, 46, 57, 60]. Worst-case iteration complexity bounds, on the other hand, have typically been overlooked when analyzing nonconvex optimization algorithms. That is, given an objective function $f : \mathbb{R}^n \to \mathbb{R}$ and a sequence of iterates $\{x_k\}$ computed for solving

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

one may ask for an upper bound on the number of iterations required to satisfy

$$\|\nabla f(x_k)\|_2 \leq \epsilon, \tag{1.2}$$

where $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$ is the gradient function of f and $\epsilon \in (0, \infty)$ is a prescribed constant. One may also go further and ask for an upper bound on the number of iterations required to satisfy a second-order stationarity condition.

The desire to design fast algorithms is obvious. Analyzing their worst-case iteration complexity, on the other hand, might seem cumbersome and unnecessary to this goal. After all, this is only one factor of the overall performance of an algorithm and it is not always clear that such worst-case analysis accurately represents the typical behavior of an algorithm. Such complexity bounds are typical in the theory of convex optimization algorithms, and are often considered seriously when evaluating such methods. There are numerous examples of scientific efforts in which the process of designing an algorithm with improved worst-case iteration complexity bound enhances the performance of the algorithm in general. One may refer to such advances in the realm of convex optimization, such as the accelerated gradient descent algorithm [53], fast iterative soft thresholding algorithm [2], and interior point methods for linear optimization. Therefore, it seems reasonable for algorithm designers to investigate complexity bounds and develop methods with improved bounds in the field of nonconvex optimization, too.

Traditionally, the most popular methods for solving (1.1) were in classes known as line search and trust region methods. Recently, however, cubic regularization methods have become popular, which are based on the pioneering work by Griewank [45] and Nesterov and Polyak [56]. Their rise in popularity is due to increased interest in algorithms with improved complexity properties, which stems from the impact of so-called optimal algorithms for solving convex optimization problems. The complexity of a traditional trust region method (e.g., see Algorithm 1 in §1.2.2) is $\mathcal{O}(\epsilon^{-2})$ (see [13] and § 1.2.2), which falls short of the $\mathcal{O}(\epsilon^{-3/2})$ complexity for cubic regularization methods (e.g., see the ARC method by [14, 15]). This latter complexity is optimal among a certain broad class of second-order methods when employed to minimize a broad class of objective functions; see [16]. That said, one can obtain even better complexity properties if higher-order derivatives are used; see [8] and [21]. The better complexity properties of regularization methods such as ARC have been a major point of motivation for discovering other methods that attain the same worst-case iteration complexity bounds.

In this work, we propose algorithms for nonconvex optimization that attain, under standard assumptions, optimal worst-case iteration complexity bounds. We begin by discussing a (nontraditional) trust region method known as TRACE (see [28] and Chapter 2) with the same optimal $\mathcal{O}(\epsilon^{-3/2})$ complexity, while at the same time allowing traditional trust region trial steps to be computed and used. We show that these properties can be

realized within the context of a trust region strategy by employing (i) modified step acceptance criteria and (ii) a novel updating mechanism for the trust region radius. Indeed, our acceptance and updating mechanisms represent significant departures from those employed in a traditional approach; e.g., there are situations in which our algorithm may *reject* a step and *expand* the trust region. A key aspect of the TRACE framework is that a solution to an implicit trust region problem is obtained by varying a regularization parameter instead of a trust region radius. This key idea has been adopted and advanced further by [5]; in particular, they propose an algorithm that has optimal iteration complexity by solving quadratic subproblems that have a carefully chosen quadratic regularization parameter.

The algorithm just mentioned is based on the assumption that an exact subproblem solution can be efficiently computed in every iteration. This is a reasonable assumption for small- to medium-scale problems, but is intractable for large-scale optimization. Thus, we extend these ideas for a general inexact regularized Newton framework to improve the practical performance of the algorithm, especially for large-scale problems. In fact, the main contributions of this extension relate to advancing the understanding of optimal complexity algorithms for solving the smooth optimization problem (1.1). Our proposed framework is intentionally very general; it is not a trust region method, a quadratic regularization method, or a cubic regularization method. Rather, we propose a generic set of conditions that each trial step must satisfy that still allow us to establish an optimal first-order complexity result as well as a second-order complexity bound similar to the methods above. Our framework contains as special cases other optimal complexity algorithms such as ARC and TRACE (see [30] and Chapter 3).

We also propose a new method for solving equality constrained nonlinear optimization problems. As is well known, such problems are important throughout science and engineering, arising in areas such as network flow optimization [48, 58], optimal allocation with resource constraints [24, 49], maximum likelihood estimations with constraints [47], and optimization with constraints defined by partial differential equations [4, 9, 59]. Contemporary methods for solving equality constrained optimization problems are predominantly based on ideas of sequential quadratic optimization (commonly known as SQP) [11, 25, 26, 35–37, 51, 57]. The design of such methods remains an active area of research as algorithm developers aim to propose new methods that attain global convergence guarantees under weak assumptions about the problem functions. Recently, however, researchers are being

drawn to the idea of designing algorithms that also offer improved worst-case iteration complexity bounds; e.g., see [17].

For solving equality constrained problems, a cubic regularization method is proposed in [19] with an eye toward achieving good complexity properties. This is a two-phase approach with a first phase that seeks an ϵ-feasible point and a second phase that seeks optimality while maintaining ϵ-feasibility. The number of iterations that the method requires in the first phase is $\mathcal{O}(\epsilon^{-3/2})$, a bound that is known to be optimal for unconstrained optimization [16]. The authors of [19] then propose a method for the second phase and analyze its complexity properties. Such a two-phase approach is analyzed further—with a careful emphasis on termination conditions for each phase—in [6]. (For related work on cubic regularization methods, see [18, 20].) Notably, the methods in [6, 19] represent a departure from the current state-of-the-art SQP methods that offer the best practical performance; see also [50]. One of the main reasons for this is that contemporary SQP methods seek feasibility and optimality *simultaneously*. By contrast, the approaches from [6, 19] might not offer practical benefits due to the fact that the first phase of each algorithm entirely ignores the objective function, meaning that numerous iterations might need to be performed before the objective function influences the trajectory of the algorithm.

Our proposed algorithm can be considered a next step in the design of *practical* algorithms for equality constrained optimization with good worst-case iteration complexity properties. Ours is also a two-phase approach, but is closer to the SQP-type methods representing the state-of-the-art for solving equality constrained problems. In particular, the first phase of our proposed approach follows a trust funnel methodology that locates an ϵ-feasible point in $\mathcal{O}(\epsilon^{-3/2})$ iterations *while also attempting to yield improvements in the objective function*. Borrowing ideas from the trust region method known as TRACE [28] (see Chapter 2), we prove that our method attains the same worst-case iteration complexity bounds as those offered by [6, 19].

1.2 Background

Consider an objective function $f : \mathbb{R}^n \to \mathbb{R}$ and a starting point $x_0 \in \mathbb{R}^n$. For the entirety of this dissertation, we assume that f is twice continuously differentiable. The main goal

in unconstrained optimization is to find a local solution x^* of problem (1.1) such that

$$f(x^*) \leq f(x), \quad \text{for all } x \in \mathbb{R}^n \text{ such that } \|x - x^*\|_2 \leq \alpha \text{ and for some } \alpha > 0.$$

Many methods have been proposed to solve problem (1.1) with global and fast local convergence guarantees. Global convergence is the ability of converging to a stationary point, which can be a first-order or second-order stationary point, from any remote initial point x_0. On the other hand, for local convergence, the rate of convergence to a second-order stationary point is considered when we are in the vicinity of such a point. Indeed, a wide variety of methods with global and fast local convergence properties have been proposed. In this section, we briefly discuss the main methods existing in the literature.

1.2.1 Newton's method

The standard Newton method is an algorithmic framework in which in every iteration, a quadratic approximation, q, of the objective function f is constructed by using second-order Taylor's expansion around the current point. The next point is then defined as a minimizer (or approximate minimizer) of the constructed model q. In other words, considering iteration k, we have:

$$x_{k+1} = x_k + s_k,$$

where

$$s_k \in \arg\min_{s \in \mathbb{R}^n} q_k(s) := f(x_k) + \nabla f(x_k)^T s + \frac{1}{2} s^T \nabla^2 f(x_k) s. \tag{1.3}$$

When $\nabla^2 f(x_k)$ is positive definite, the subproblem (1.3) is well-defined and there exists a closed-form expression for a Newton iteration as the following:

$$x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \nabla f(x_k).$$

Newton's method in its original format as stated in (1.3) is not globally convergent, even for strongly convex problems. Moreover, the solution of subproblem (1.3) is not well-defined when $\nabla^2 f(x)$ is indefinite or negative definite. On the other hand, under some standard assumptions, such as Lipschitz continuity of the Hessian function $\nabla^2 f(x)$ for all x sufficiently close to a nondegenerate second-order stationary point x^* such that $\nabla^2 f(x^*)$

is positive definite, Newton's method enjoys a quadratic rate of convergence [31]. This fast local convergence behavior, despite the fact that Newton's method is not globally convergent, has motivated much research around modifying this method to globalize its convergence properties, such as using Hessian modifications combined with a line search, a trust region methodology, or a regularization approach.

1.2.2 Trust region methods

As mentioned in the last subsection, Newton's algorithm suffers from the possibility of divergence, but the fast local convergence of this algorithm makes it attractive to explore ways to globalize the method while maintaining its local convergence behavior. Over the years, many variants of Newton's method with globalization strategies have been developed. For instance, one very popular approach has been to combine line search ideas and Hessian modifications, the latter of which ensures positive definiteness of the Hessian matrices so that the subproblems are well-defined. Another popular approach has been to incorporate trust regions, in which a norm constraint is added to the subproblem to make it well defined, even if the Hessian is not positive definite.

In line search methods, the search direction is chosen or computed first, then the step size is decided, usually based on Armijo-Wolfe conditions [57]. In trust region methods, on the contrary, a region, typically spherical in shape, is defined in which the approximation of the objective function can be "trusted". Within this "trust region", a direction is computed that minimizes the model. In a standard trust region method, the trust region is defined by its radius, δ, such that $\|s\|_2 \leq \delta$; therefore, the trust region subproblem at iteration k is

$$\begin{aligned} s_k \in \min_{s \in \mathbb{R}^n} \; & q_k(s) = f(x_k) + \nabla f(x_k)^T s + \frac{1}{2} s^T \nabla^2 f(x_k) s \\ \text{s.t.} \; & \|s\| \leq \delta_k. \end{aligned} \tag{1.4}$$

To ensure the convergence of the algorithm, there are some rules used to update the trust region radius. Any reasonable updating procedure has to consider the quality of the step computed in each iteration; therefore, defining a proper measure of solution quality is critical for the convergence of this algorithm. One popular quality measure in the literature constructs the ratio of "actual reduction" over "predicted reduction" of the objective value, called ρ_k at iteration k, and compares the resulting ratio with a pre-specified constant η. If

$\rho_k \geq \eta$, then the trial step computed by solving (1.4) will be accepted; otherwise, the trial step will be rejected. In the latter case, an update in δ is necessary to generate a different solution in the next step. As the Taylor approximation is more accurate within a smaller region, a proper update is to decrease the trust region radius for the next iteration. While not necessary, δ will be increased for the case $\rho_k \geq \eta$, with the hope of faster convergence. Algorithm 1 is a common representation of a standard trust region method [23].

Algorithm 1 Trust Region Method

Require: an acceptance constant $\eta \in \mathbb{R}_{++}$ with $0 < \eta < 1$
Require: update constants $\{\gamma_{\text{C}}, \gamma_{\text{E}}\} \subset \mathbb{R}_{++}$ with $0 < \gamma_{\text{C}} < 1 < \gamma_{\text{E}}$

choose $x_0 \in \mathbb{R}^n$, $\delta_0 \in \mathbb{R}_{++}$
for $k = 0, 1, 2, \ldots$ **do**
    compute s_k by solving (1.4)
    compute ρ_k as the following:

$$\rho_k := \frac{f(x_k) - f(x_k + s_k)}{f(x_k) - q_k(s_k)} \tag{1.5}$$

    if $\rho_k \geq \eta$ **then** [accept step and expand trust region]
        set $x_{k+1} \leftarrow x_k + s_k$
        set $\delta_{k+1} \leftarrow \max\{\delta_k, \gamma_{\text{E}} \|s_k\|_2\}$
    else ($\rho_k < \eta$) [reject step and contract trust region]
        set $x_{k+1} \leftarrow x_k$
        set $\delta_{k+1} \leftarrow \gamma_{\text{C}} \|s_k\|_2$

Under some standard assumptions, such as Lipschitz continuity of the gradient function ∇f, Algorithm 1 enjoys global convergence. In addition, under the assumption of Lipschitz continuity of $\nabla^2 f$ for all x sufficiently close to a nondegenerate second-order stationary point x^* such that $\nabla^2 f(x^*)$ is positive definite, this algorithm has quadratic local convergence, similar to Newton's method [57].

1.2.2.1 Worst-case first-order complexity of trust region methods

Despite the good performance that people have seen for trust region methods, in terms of their complexity, one can show that they might require $\mathcal{O}(\epsilon^{-2})$ iterations to find an ϵ-stationary point. In fact, an example, constructed by Cartis, Gould, and Toint [13], shows that for any $\tau > 0$, Newton's method needs at least $\mathcal{O}(\epsilon^{-2+\tau})$ iterations to achieve the stationarity measure tolerance (1.2). The same example can be used to show that the traditional trust region Algorithm 1 follows the same steps similar to those generated by

Newton's method [16].

Consider the following two-dimensional example in which the sequence $\{x_k\}$ is the one that would be generated by Newton's method:

$$x_0 = (0,0)^T, \quad x_{k+1} = x_k + s_k, \quad s_k = \begin{pmatrix} \left(\frac{1}{k+1}\right)^{\frac{1}{2}+\bar{\tau}} \\ 1 \end{pmatrix}, \tag{1.6a}$$

$$f(x_0) = \frac{1}{2}[\zeta(1+2\bar{\tau}) + \zeta(2)], \quad f(x_{k+1}) = f(x_k) - \frac{1}{2}\left[\left(\frac{1}{k+1}\right)^{1+2\bar{\tau}} + \left(\frac{1}{k+1}\right)^2\right], \tag{1.6b}$$

$$\nabla f(x_k) = -\begin{pmatrix} \left(\frac{1}{k+1}\right)^{\frac{1}{2}+\bar{\tau}} \\ \left(\frac{1}{k+1}\right)^{\frac{1}{2}} \end{pmatrix}, \tag{1.6c}$$

and

$$\nabla^2 f(x_k) = \begin{pmatrix} 1 & 0 \\ 0 & \left(\frac{1}{k+1}\right)^2 \end{pmatrix}, \tag{1.6d}$$

where $\bar{\tau} = \tau/(4-2\tau) > 0$ and $\zeta(t) := \sum_{k=1}^{\infty} k^{-t}$ is the Riemann ζ function [13]. With the sequence of gradients as in (1.6c), the number of iterations needed to satisfy the stationarity measure tolerance (1.2) is at least $\mathcal{O}(\epsilon^{-2+\tau})$. For the defined s_k, $\nabla f(x_k)$, and $\nabla^2 f(x_k)$, we have

$$\nabla^2 f(x_k) s_k = -\nabla f(x_k),$$

with positive definite $\nabla^2 f(x_k)$; therefore, s_k globally minimizes $q_k(s)$. Furthermore,

$$f(x_k + s_k) = q_k(s_k),$$

which guarantees $\rho_k = 1$; therefore, according to Step 5, the trial step s_k will be accepted. The definition of s_k in (1.6a) guarantees that $\|s_{k+1}\|_2 < \|s_k\|_2$ for all $k \geq 0$. On the other hand, from $\rho_k = 1$, Step 5, and Step (7), one can deduce that $\delta_{k+1} \geq \delta_k$ for all $k \geq 0$. If $\delta_0 \geq \|s_0\|_2 = \sqrt{2}$, then the sequence of $\{x_k\}$ defined in (1.6a) can be reached by applying the traditional trust region Algorithm 1 to minimizing a function satisfying (1.6); therefore, Algorithm 1 requires at least $\mathcal{O}(\epsilon^{-2+\tau})$ iterations to achieve the stationarity measure tolerance (1.2).

To find a function satisfying (1.6), Cartis, Gould, and Toint [13] have used polynomial

Hermite interpolation on the interval $[0, x_{k+1} - x_k]$. To do so, consider

$$f(x) = f_1([x]_1) + f_2([x]_2), \tag{1.7}$$

where $[x]_i$ is the i-th component of x for $i \in \{1, 2\}$. Using Hermite interpolation, for all $k \geq 0$ we have

$$f_1([x]_1) = p_k([\hat{x} - x_k]_1) + f_1([x_{k+1}]_2) \;\text{ for }\; [\hat{x}]_1 \in [[x_k]_1, [x_{k+1}]_1]\,, \tag{1.8}$$

where p_k is the polynomial

$$p_k(s) = c_{0,k} + c_{1,k}s + c_{2,k}s^2 + c_{3,k}s^3 + c_{4,k}s^4 + c_{5,k}s^5, \tag{1.9}$$

with coefficients defined to satisfy the interpolation conditions

$$\begin{aligned}
&p_k(0) = \frac{1}{2}\left(\frac{1}{k+1}\right)^{1+2\bar{\tau}}, && p_k([s_k]_1) = 0;\\
&\nabla p_k(0) = -\left(\frac{1}{k+1}\right)^{\frac{1}{2}+\bar{\tau}}, && \nabla p_k([s_k]_1) = -\left(\frac{1}{k+2}\right)^{\frac{1}{2}+\bar{\tau}};\\
&\nabla^2 p_k(0) = 1, && \nabla^2 p_k([s_k]_1) = 1.
\end{aligned} \tag{1.10}$$

In addition, for the univariate function f_2 we have for all $k \geq 0$

$$f_2([x]_2) = w_k([\hat{x} - x_k]_2) + f_1([x_{k+1}]_2) \;\text{ for }\; [\hat{x}]_2 \in [[x_k]_1, [x_{k+1}]_2]\,, \tag{1.11}$$

where w_k is the polynomial

$$w_k(s) = d_{0,k} + d_{1,k}s + d_{2,k}s^2 + d_{3,k}s^3 + d_{4,k}s^4 + d_{5,k}s^5, \tag{1.12}$$

with coefficients defined to satisfy the interpolation conditions

$$\begin{aligned}
&w_k(0) = \frac{1}{2}\left(\frac{1}{k+1}\right)^{2}, && w_k(1) = 0;\\
&\nabla w_k(0) = -\left(\frac{1}{k+1}\right)^{2}, && \nabla w_k(1) = -\left(\frac{1}{k+2}\right)^{2};\\
&\nabla^2 w_k(0) = \left(\frac{1}{k+1}\right)^{2}, && \nabla^2 w_k(1) = \left(\frac{1}{k+2}\right)^{2}.
\end{aligned} \tag{1.13}$$

Therefore, according to [13]

$$
\begin{aligned}
c_{0,k} &= \frac{1}{2}\left(\frac{1}{k+1}\right)^{1+2\bar{\tau}}, \quad c_{1,k} = -\left(\frac{1}{k+1}\right)^{\frac{1}{2}+\bar{\tau}}, \quad c_{2,k} = \frac{1}{2} \\
c_{3,k} &= 4\left(\frac{(k+1)^2}{k+2}\right)^{\frac{1}{2}+\bar{\tau}} \\
c_{4,k} &= -7\left(\frac{(k+1)^3}{k+2}\right)^{\frac{1}{2}+\bar{\tau}} \\
c_{5,k} &= 3\left(\frac{(k+1)^4}{k+2}\right)^{\frac{1}{2}+\bar{\tau}}.
\end{aligned}
\tag{1.14}
$$

Furthermore, according to [13]

$$
\begin{aligned}
d_{0,k} &= \frac{1}{2}\left(\frac{1}{k+1}\right)^2, \quad d_{1,k} = -\left(\frac{1}{k+1}\right)^2, \quad d_{2,k} = \frac{1}{2}\left(\frac{1}{k+1}\right)^2 \\
d_{3,k} &= \frac{9}{2}\left(\frac{1}{k+2}\right)^2 - \frac{1}{2}\left(\frac{1}{k+1}\right)^2 \\
d_{4,k} &= -8\left(\frac{1}{k+2}\right)^2 + \left(\frac{1}{k+1}\right)^2 \\
d_{5,k} &= \frac{7}{2}\left(\frac{1}{k+2}\right)^2 - \left(\frac{1}{k+1}\right)^2.
\end{aligned}
\tag{1.15}
$$

The proofs of Lipschitz continuity of $\nabla f(x)$ and $\nabla^2 f(x)$ and boundedness of f are presented in [13]; therefore, the function f constructed this way satisfies all the assumptions needed for convergence of Algorithm 1. This example shows that trust region methods can be inefficient, in the sense that they might take $\mathcal{O}(\epsilon^{-2})$ iterations to achieve the stationarity tolerance (1.2). In the next subsection, we turn to an alternative framework that achieves improved worst-case iteration complexity over trust region methods.

1.2.3 Cubic regularization methods

As mentioned, worst-case iteration complexity bounds have typically been overlooked when analyzing nonconvex optimization algorithms. This situation in the field of nonconvex optimization has started to change with recent studies on cubic regularization methods. Originally proposed in a technical report by Griewank [45], the foundation for

worst-case iteration complexity bounds for second-order methods using cubic regularization was first established in the seminal work by Nesterov and Polyak [56]. (See also [61] for another early article on such methods.) Nesterov and Polyak [56] proposed and analyzed an abstract algorithm called Cubic regularization of Newton Method (or CNM as referred to in [55]). In this algorithm, to overcome issues related to ill-defined subproblems when the Hessian is indefinite, a cubic term is added to the quadratic model $q_k(s)$. The resulting subproblem is an unconstrained (potentially) nonconvex cubic problem

$$s_k \in \arg\min_{s\in\mathbb{R}^n} c_k(s) := f(x_k) + \nabla f(x_k)^T s + \frac{1}{2} s^T \nabla^2 f(x_k) s + \frac{\sigma_k}{6}\|s\|_2^3,$$

where σ_k is the algorithm parameter whose value is set in every iteration to satisfy a so-called sufficient decrease property defined as

$$f(x_k + s_k) \leq c(s_k). \tag{1.16}$$

The inequality (1.16) is satisfied for every $\sigma_k \geq L$, where L is the Lipschitz constant of the Hessian function $\nabla^2 f(x)$; therefore, a procedure of increasing σ_k by multiplying with a constant will eventually terminate after satisfying (1.16). Algorithm 2 shows the original CNM algorithm from [56].

Algorithm 2 Cubic regularization of Newton Method (CNM)

Require: a constant $L_0 \in \mathbb{R}_{++}$ with $L_0 \leq L$

1: choose $x_0 \in \mathbb{R}^n$
2: **for** $k = 0, 1, 2, \ldots$ **do**
3: find $\sigma_k \in [L_0, 2L]$ such that (1.16) is satisfied
4: set $x_{k+1} \leftarrow x_k + s_k$

The global convergence of Algorithm 2 to a second-order stationary point is proved under the assumption of Lipschitz continuity of ∇f and $\nabla^2 f$. While the former assumption is pretty standard in the literature for convergence, the latter one seems strong because it assumes global Lipschitz continuity of the Hessian, not just near a minimizer. But, one should notice that they have proved the convergence to a local solution x^* satisfying the second-order sufficient condition, that is $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semidefinite. In addition, there exist several lemmas and theorems in their paper from which the worst-case iteration complexity bound of $\mathcal{O}(\epsilon^{-3/2})$ to a first-order stationary point and $\mathcal{O}(\epsilon^{-3})$

to a second-order stationary point can be inferred for successful (those satisfying (1.16)) iterations.

Despite the salient properties of Algorithm 2, it is far away from being a practical algorithm. First of all, the Lipschitz continuity of the Hessian function might be a strong assumption for those who only need the convergence to a stationary point with zero gradient. Second, the assumption of knowing the Lipschitz constant L beforehand to use in the algorithm is extremely restrictive. Last but not least, finding $\sigma_k \in [L_0, 2L]$ satisfying (1.16) as the way stated in Algorithm 2 might not be efficient in terms of complexity bounds.

More recently, the Adaptive Regularisation framework using Cubics, also known as the ARC algorithm [14, 15], was developed with an eye towards practical implementations. In this work as in [56], Cartis, Gould, and Toint propose an algorithm in which the trial steps are computed by minimizing a local cubic model of the objective function at each iterate. The expense of this computation is similar to that of solving a subproblem arising in a typical trust region method, and their overall algorithm—which has essentially the same flavor as a trust region method—is able to attain global and fast local convergence guarantees. However, the distinguishing feature of ARC and other cubic regularization algorithms is that, under reasonable assumptions, they ensure that the stationarity measure tolerance (1.2) is guaranteed to hold after at most $\mathcal{O}(\epsilon^{-3/2})$ iterations. Furthermore, the analysis in [16] shows that the complexity bound for ARC is optimal with respect to a particular class of second-order methods for minimizing a particular class of sufficiently smooth objective functions in the optimization problem (1.1).

In ARC, the subproblems are similar to CNM, where a cubic regularization of a second-order Taylor series will be solved, namely

$$s_k \in \arg\min_{s\in\mathbb{R}^n} C_k(s) := f(x_k) + \nabla f(x_k)^T s + \frac{1}{2} s^T B_k s + \frac{\sigma_k}{3}\|s\|_2^3, \tag{1.17}$$

where B_k is an estimation of $\nabla^2 f(x_k)$ and σ_k is the algorithm parameter whose value is updated dynamically by the algorithm. To measure the quality of the trial step s_k, a ratio of actual reduction over predicted reduction is computed as the following:

$$\rho_k^C := \frac{f(x_k) - f(x_k + s_k)}{f(x_k) - C_k(s_k)}. \tag{1.18}$$

Then, based on this ratio the acceptance or rejection of the trial step is decided, moreover, σ_{k+1} is set properly. Algorithm 3 shows the ARC algorithm.

Algorithm 3 Adaptive Regularization using Cubics (ARC)

Require: two acceptance constants $\{\eta_1, \eta_2\} \subset \mathbb{R}_{++}$ with $0 < \eta_1 < \eta_2 < 1$
Require: update constants $\{\gamma_1, \gamma_2\} \subset \mathbb{R}_{++}$ with $1 < \gamma_1 < \gamma_2$

choose $x_0 \in \mathbb{R}^n$, $\delta_0 \in \mathbb{R}_{++}$
for $k = 0, 1, 2, \ldots$ **do**
    compute s_k by solving (1.17)
    compute ρ_k^C using (1.18)
    if $\rho_k^C \geq \eta_1$ **then**
        set $x_{k+1} \leftarrow x_k + s_k$
        if $\rho_k^C \geq \eta_2$ **then**
            set $\sigma_{k+1} \in [0, \sigma_k]$
        else $(\eta_1 < \rho_k^C < \eta_2)$
            set $\sigma_{k+1} \in [\sigma_k, \gamma_1 \sigma_k]$
    else $(\rho_k^C < \eta)$
        set $x_{k+1} \leftarrow x_k$
        set $\sigma_{k+1} \in [\gamma_1 \sigma_k, \gamma_2 \sigma_k]$

Because of the importance of the ARC algorithm in the development and analysis of our proposed algorithms described in later chapters, a more official statement of the convergence properties and the complexity bounds of this algorithm is presented in the following theorem.

Theorem 1. *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and the sequence $\{f_k\}$ is bounded from below. If the gradient function is Lipschitz continuous and B_k is bounded in norm for all k, then $\{\nabla f(x_k)\} \to 0$. Furthermore, if f is twice continuously differentiable, B_k is a relatively close approximation of $\nabla^2 f(x_k)$ such that $\|B_k - \nabla^2 f(x_k)\|_2 \to 0$ whenever $\|\nabla f(x_k)\|_2 \to 0$ and $k \to \infty$, and there exists a subsequence of iterates converging to x^* with positive definite $\nabla^2 f(x^*)$, then the whole sequence $\{x_k\}$ converges to x^*. In addition, if $\nabla^2 f(x)$ is locally Lipschitz continuous close to x^*, $\|(B_k - \nabla^2 f(x_k))s_k\|_2 \leq C\|s_k\|_2^2$, for all k, and some constant $C > 0$, and $\sigma_k \geq \sigma_{min}$ for all k and some constant $\sigma_{min} > 0$, then $x_k \to x^*$ and $\nabla f(x_k) \to 0$ Q-quadratically. Moreover, if $\nabla^2 f(x)$ is globally Lipschitz continuous on the path of computed iterates, then the whole sequence $\{x_k\}$ converges to x^* with $\nabla f(x^*) = 0$ and positive semidefinite $\nabla^2 f(x^*)$ from any remote initial point x_0.*

With the same assumptions, for any $\bar{\epsilon} > 0$, the stationarity measure tolerance (1.2) is guaranteed to hold after at most $\mathcal{O}(\epsilon^{-3/2})$ iterations, where $\epsilon \in (0, \bar{\epsilon}]$. In addition,

the number of iterations required to find a point x_k such that the smallest eigenvalue of $\nabla^2 f(x_k)$ is greater than or equal to $-\epsilon$ is at most $\mathcal{O}(\epsilon^{-3})$.

Although in Step 3 of Algorithm 3 the global solution of (1.17) is considered as s_k, for Theorem 1 to be valid, such a solution is not required, but s_k needs to be a global solution for (1.17) on a subspace containing the gradient $\nabla f(x_k)$ with

$$\|\nabla C_k(s_k)\|_2 \leq \theta \|\nabla f(x_k)\|, \tag{1.19}$$

for some constant $\theta > 0$. This milder requirement makes ARC more practical especially for large-scale problems where solving the nonconvex problem (1.17) might be prohibitively expensive. Under this more relaxed assumption, s_k must satisfy the following condition:

$$\nabla f(x_k)^T s_k + s_k^T B_k s_k + \sigma_k \|s_k\|^3 = 0, \tag{1.20}$$

(which is equivalent to $\nabla C_k(s_k)^T s_k = 0$) and

$$s_k^T B_k s_k + \sigma_k \|s_k\|^3 \geq 0. \tag{1.21}$$

The importance of the improvement in the complexity bound gets more noticeable when it is compared to the trust region method. As one may notice that Algorithms 1 and 3 are very similar in terms of the framework. In fact, the similarity of these methods are not restricted only to the framework; the subproblem solutions can be the same if one chooses the appropriate parameters. For example, the solution s_k of (1.17) is also an optimal solution for (1.4) if δ_k is set to $\|s_k\|_2$. Furthermore, an optimal solution s_k for (1.4) is also an optimal solution for (1.17) with $\sigma_k = \lambda_k / \|s_k\|_2$, where λ_k is the dual variable associated with the trust region constraint.

Despite the improved worst-case iteration complexity bounds of ARC, there is not a noticeable performance improvement versus trust region method, yet in many cases, the trust region algorithm, even in its standard version, beats the ARC algorithm in terms of solution time and even iterations needed to satisfy a measure of convergence as (1.2). One of the main motivations of this dissertation is to answer the question if there exists a trust region method with improved iteration complexity bounds as those of ARC. In the next chapter, we answer this question.

Our motivation of proposing a new trust region algorithm is to introduce an algorithm which can perform well in practice while theoretically achieves improved complexity bounds. To this end, an inexact regularized Newton framework is proposed in Chapter 3, which allows inexact Newton steps whenever possible while achieves the improved worst-case iteration complexity bounds, similar to those of ARC and TRACE.

1.2.4 Methods for equality constrained problems

In real world problems, the decision variable x in problem (1.1) might need to be restricted, e.g., due to resource limitations, state and federal regularizations, technology bounds, and engineering designs. Therefore, many optimization problems are "constrained" problems of the form

$$\min_{x \in \mathcal{X}} f(x), \tag{1.22}$$

where $\mathcal{X}$ is the set of all possible decisions defined by constraints. The set $\mathcal{X}$ in (1.22) might possess different properties; e.g., it might be convex or not convex, closed or open, connected or disconnected, etc. In this work, instead of the general form (1.22), we assume that $\mathcal{X}$ can be defined by a multivariate continuous function $c : \mathbb{R}^n \to \mathbb{R}^m$ such that $\mathcal{X} = \{x \in \mathbb{R}^n \mid c(x) = 0\}$; therefore, we assume that $\mathcal{X}$ is closed. For some algorithms, there might be other assumptions on c such as differentiability, smoothness, and so-called constraints qualifications. The constrained problem, then, will be defined as the following:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & c(x) = 0. \end{aligned} \tag{1.23}$$

The widespread application of (1.23) has motivated an enormous deal of research to develop "reliable" algorithms to solve it. The term "reliable" itself needs clarification; for us, a very loose definition of a reliable algorithm can be expressed as an algorithm with the ability to find a point x^* or generate a sequence of points $\{x_k\}$ converging to x^* that satisfies the Karush-Kuhn-Tucker (KKT) optimality conditions of (1.23) or, in the case of an infeasible instance of (1.23), a sequence converging to a stationary point c. Once again, the main body of research for many years was focused on designing algorithms with global convergence properties and, similar to the unconstrained case, the analysis of the worst-case iteration complexity bounds is a recent trend on this area; e.g., see the recent

work in [18–20].

Before going any further, let us define the KKT optimality conditions. The Lagrangian function of (1.23) is defined as $\mathcal{L}(x, y) := f(x) + c(x)^T y$. Let f and c be continuously differentiable. The pair (x, y) is a KKT point if:

$$\begin{aligned} \nabla f(x) + \nabla c(x)^T y &= 0 \\ c(x) &= 0. \end{aligned} \tag{1.24}$$

KKT conditions are necessary for any optimal (local or global) solution if some sort of constraint qualification, such as the Linear Independence constraint qualification (LICQ) [57], holds. Similar to the unconstrained case, there are also second-order optimality conditions given f and c are twice continuously differentiable.

1.2.4.1 Sequential quadratic programming

Similar to unconstrained optimization algorithms, one may consider to iteratively solve a model of problem (1.23) and take the step found by solving the model. Sequential Quadratic Programming/Optimization (SQP) is a well-known iterative framework in which a second-order model of the objective function and a linearization of the constraint functions are considered as a model of the original problem (1.23), leading to a subproblem of the form

$$\begin{aligned} \min_{s \in \mathbb{R}^n} \quad & f(x) + \nabla f(x)^T s + \frac{1}{2} s^T B s \\ \text{s.t.} \quad & c(x) + \nabla c(x) s = 0, \end{aligned} \tag{1.25}$$

where B is positive definite. The optimality conditions (KKT conditions) for problem (1.25) are

$$\begin{aligned} \nabla f(x) + Bs + \nabla_x c(x)^T \eta &= 0 \\ c(x) + \nabla_x c(x) s &= 0, \end{aligned} \tag{1.26}$$

where η is an $m \times 1$ vector of dual variables for problem (1.25). We can rewrite these linear equations in matrix form as

$$\begin{bmatrix} B & \nabla c(x)^T \\ \nabla c(x) & 0 \end{bmatrix} \begin{bmatrix} s \\ \eta \end{bmatrix} = - \begin{bmatrix} \nabla f(x) \\ c(x) \end{bmatrix}. \tag{1.27}$$

If one applies Newton's method to solve (1.24), one will have the following linear system of equations:

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x,y) & \nabla c(x)^T \\ \nabla c(x) & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = - \begin{bmatrix} \nabla f(x) + \nabla c(x)^T y \\ c(x) \end{bmatrix}. \tag{1.28}$$

A closer look to two equations (1.28) and (1.27) shows that these two equations are exactly the same if we set $B = \nabla^2_{xx}\mathcal{L}(x,y)$ and $\eta = y + \Delta y$. This observation shows that solving the problem (1.23) by using SQP is exactly the same as solving the KKT conditions (1.24) by using Newton's method; therefore, the SQP algorithm in its original form is not globally convergent. Hence, combining other globalization frameworks such as line search or trust region strategies is required. In addition, to be able to analyze the SQP method, assuming that the KKT conditions are necessary is a standard assumption. This can be done, e.g., by making the following assumption.

Assumption 1. *The matrix $\nabla c(x^*)$ has full row rank for any $x^* \in \arg\min_{x\in\mathcal{X}} f(x)$.*

Assumption 1 means that the LICQ is satisfied at x^*; so the KKT conditions are necessary for any minimizer of (1.23).

Another typical assumption is the following.

Assumption 2. *It holds that $d^T\nabla^2_{xx}\mathcal{L}(x,y)d \geq 0$ for all d such that $\nabla c(x)^T d = 0$.*

Assumption 2 is useful to guarantee local convergence when we are in a vicinity of x^*, because positive definiteness of $\nabla^2_{xx}\mathcal{L}(x,y)$ is essential for fast local convergence of SQP.

1.2.4.2 SQP with a trust region constraint

In the case of SQP, similar to the unconstrained case for Newton's method, to make the algorithm globally convergent, using a line search, trust region framework, or other globalization mechanism is required. Here, we discuss the use of trust regions to globalize the method. In trust region methods, we try to find a solution for the model within a "trusted" area which the original function is believed to behave almost the same as the estimated function. In our present case, this leads to a subproblem of the form

$$\begin{aligned} \min_{s\in\mathbb{R}^n} \quad & f(x) + \nabla f(x)^T s + \frac{1}{2} s^T B s \\ \text{s.t.} \quad & c(x) + \nabla c(x) s = 0 \\ & \|s\| \leq \delta, \end{aligned} \tag{1.29}$$

where δ is the trust region radius.

A challenge in constrained optimization is the possibility of achieving an inconsistent subproblem by adding a trust region constraint. This will make the algorithm not well defined unless modifications are made. Figure 1.1 shows such a subproblem in a two-dimensional space.

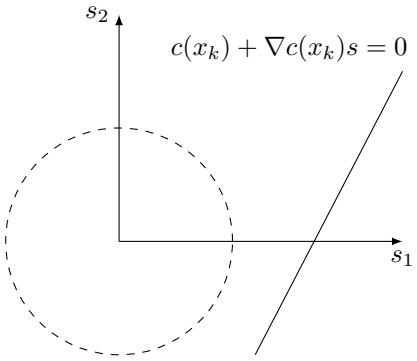


Figure 1.1: Inconsistent trust region constraint

Several tricks have been proposed to handle this problem [57], but among all of them, we will discuss the step decomposition strategy. In this procedure, the step s_k is decomposed into two components: n_k and t_k (where n_k should not be confused with the number of variables n). The first one, n_k is chosen to find the minimum value of $\|c(x_k) + \nabla c(x_k) n_k\|_2^2$ within the trust region (or as it is the most common, into a region smaller than the original trust region). Then, t_k is chosen in the null space of $\nabla c(x_k)$,

$\mathcal{N}(\nabla c(x_k)) := \{x \in \mathbb{R}^n \mid \nabla c(x_k)x = 0\}$, to solve a possibly modified version of (1.29):

$$\begin{aligned} \min_{s\in\mathbb{R}^n} \quad & f(x_k) + \nabla f(x_k)^T s + \frac{1}{2} s^T B_k s \\ \text{s.t.} \quad & c(x_k) + \nabla c(x_k)s = c(x_k) + \nabla c(x_k)n_k \\ & \|s\| \leq \delta, \end{aligned} \tag{1.30}$$

where $s_k = n_k + t_k$. Although not always necessary, n_k is commonly assumed to be in the range space of $\nabla c(x_k)^T$, $\mathcal{R}(\nabla c(x_k)^T) := \{\nabla c(x_k)^T x \mid x \in \mathbb{R}^m\}$, which is why it is called the "normal" step. In addition, t_k is called the "tangential" step as it is in the null space of $\nabla c(x_k)$. (Notice that for any matrix A, the range space of A^T and the null space of A are orthogonal spaces.)

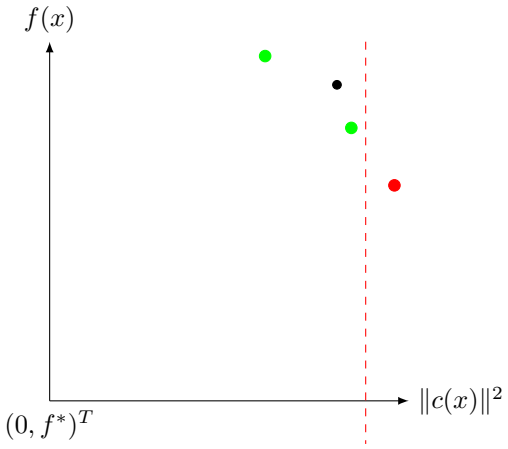
Another issue that we have to consider in designing a trust region SQP algorithm is the definition of a good measure for progress such as a merit function. In unconstrained problems, the natural merit function is simply f itself, but for constrained problems, a merit function has to consider feasibility as well as the objective function value. For a complete review of these issues and the possible solutions, one may refer to numerous references such as [57].

The next two subsections are dedicated to two globally convergent algorithms for solving (1.23) with two completely different approaches to solve these above-mentioned issues and also with two different design objectives. The trust funnel algorithm is a method based on a step decomposition strategy in which the main goal is to generate a globally convergent algorithm. The Short Step ARC (ShS-ARC) algorithm, on the other hand, is designed to achieve not only global convergence but also to guarantee good worst-case iteration complexity bounds for solving constrained problems. In this algorithm, subproblems are unconstrained, so inconsistency of the subproblems is not an issue.

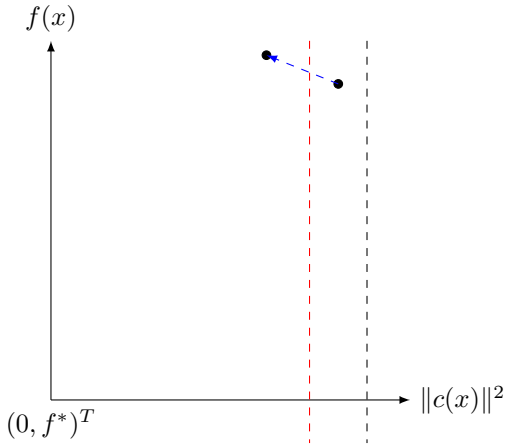
1.2.4.3 Trust funnel algorithm

A recent algorithmic framework with global convergence guarantees [27, 38] based on a step decomposition strategy will be discussed here. We present this algorithm in details as it will be used in our proposed new algorithm with good worst-case iteration complexity bounds in Chapter 4. The trust funnel algorithm attempts to drive a measure of infeasibility to zero at the same time that it tries to improve the objective value. In particular,

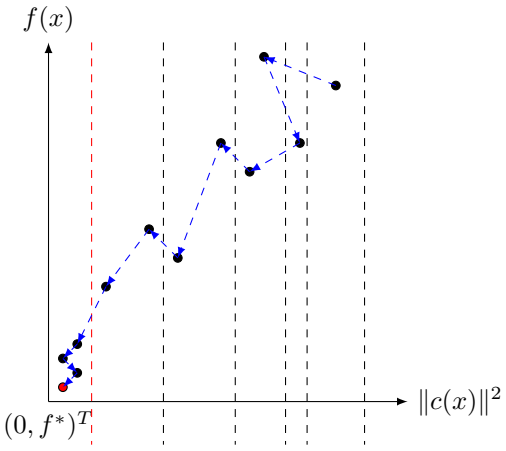
progress towards reducing the constraint violation measures is guided by a tolerance that is reduced dynamically by the algorithm. (See Figure 1.2.)



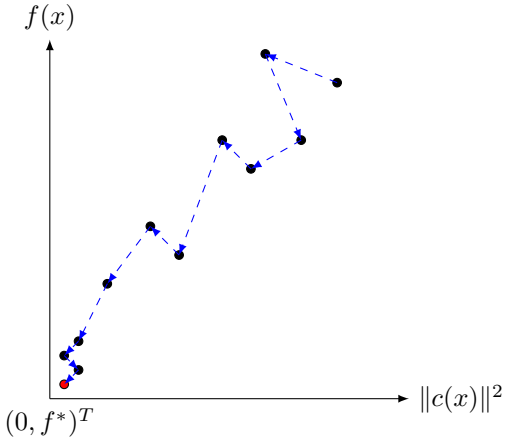
(a) The funnel condition: The algorithm is not allowed to take steps on the right side of the dashed line. The green points could be accepted while the red one would not.



(b) Updating the funnel radius: Whenever the algorithm takes a V-ITERATION (a step toward the feasible region), the funnel radius gets updated to create a tighter region.



(c) Finding an approximate solution: The algorithm generates a sequence of steps while updating the funnel radius until an approximate solution is found.



(d) The funnel algorithm: The algorithm drives a measure of infeasibility toward zero while improving the objective function at the same time.

Figure 1.2: Illustration of the trust funnel algorithm

For equality constrained problems, a reasonable measure of infeasibility can be defined as the sum of squares of the constraint values, or equivalently, a function $v : \mathbb{R}^n \to \mathbb{R}$ is defined which measures the infeasibility of a point x as the following:

$$v(x) := \frac{1}{2}\|c(x)\|_2^2. \tag{1.31}$$

During the running of the algorithm, the hope is that the value of v converges to zero while the solution approaches to a stationary point of the Lagrangian function

$$\mathcal{L}(x, y) := f(x) + y^T c(x); \tag{1.32}$$

otherwise the algorithm fails in finding a first-order stationary point of the problem. In addition, we define $v^{\max}$ as a dynamic parameter, whose value is set in each iteration such that $v_{k+1}^{\max} \leq v_k^{\max}$. The trial step s_k can be accepted only if $v(x_k + s_k) \leq v_k^{\max}$, which is called the funnel condition.

In a typical trust funnel algorithm, three vectors are computed in each iteration k: the normal step n_k, the tangential step t_k, and the multiplier estimate y_k. The vector n_k is the step taken to minimize the Gauss-Newton approximation of the function v, subject to a trust region constraint, defined as the following:

$$\begin{aligned} n_k \in \arg\min_{n \in \mathbb{R}^n} \quad & m_k^v(n) := \frac{1}{2}\|c(x_k) + \nabla c(x_k) n\|_2^2 \\ \text{s.t.} \quad & \|n\|_2 \leq \delta_k^v, \end{aligned} \tag{1.33}$$

where δ_k^v is the trust region radius for normal step at iteration k. After finding n_k, an estimate of the Lagrangian multiplier y_k is computed:

$$y_k \in \arg\min_{y \in \mathbb{R}^m} \ \frac{1}{2}\|\nabla f(x_k) + \nabla^2 f(x_k) n_k + \nabla c(x_k)^T y\|_2^2. \tag{1.34}$$

Problem (1.34) tries to find multiplier y_k such that the difference of $-\nabla f(x_k) - \nabla^2 f(x_k) n_k$ (an approximation of the negative gradient of the objective at $x_k + n_k$) and $\nabla c(x_k)^T y_k$ gets the minimum possible value. This, in turn, means that y_k is found such that $\nabla c(x_k)^T y_k$ is the projection of $-\nabla f(x_k) - \nabla^2 f(x_k) n_k$ onto $\mathcal{R}(\nabla c(x_k)^T)$. Hence, $\nabla f(x_k) + \nabla^2 f(x_k) n_k + \nabla c(x_k)^T y_k$ is the projection of $-\nabla f(x_k) - \nabla^2 f(x_k) n_k$ onto $\mathcal{N}(\nabla c(x_k))$. To justify the appropriateness of estimating y_k from (1.34) one may notice that at a first-order stationary

point x_k, KKT conditions are satisfied for the pair (x_k, y_k), where y_k is computed from (1.34).

After computing n_k and y_k, the tangential step t_k is determined to solve the following problem:

$$\begin{aligned} t_k \in \arg\min_{t\in\mathbb{R}^n} \ & m_k^f(n_k+t) := f(x_k) + \nabla c(x_k)^T(n_k+t) + \frac{1}{2}(n_k+t)^T\nabla^2 f(x_k)(n_k+t) \\ \text{s.t.} \ & \nabla c(x)t = 0 \\ & \|n_k+t\|_2 \le \delta_k^s, \end{aligned} \tag{1.35}$$

where δ_k^s is the radius of the region within which both constraint model and objective function model can be trusted. To this end, one may define $\delta_k^s := \min\{\kappa_v\delta_k^v, \delta_k^f\}$, where $\kappa_v \in \mathbb{R}_{++}$ with $\kappa_v > 1$ and δ_k^f is the trust region radius for objective model. Another reason for using (1.34) to estimate y_k relates to the fact that $\nabla f(x_k) + \nabla^2 f(x_k)n_k + \nabla c(x_k)^T y_k$ is in $\mathcal{N}(\nabla c(x_k))$; thus, this vector can be used as an initial search direction for (1.35). In fact, for convergence purposes there is no need to solve (1.35) exactly, while finding a solution which is at least as good as the optimal solution in the direction of $\nabla f(x_k) + \nabla^2 f(x_k)n_k + \nabla c(x_k)^T y_k$ (sometimes referred to as the Cauchy direction) suffices.

After finding the normal step, new multiplier, and tangential step, the algorithm decides to perform a so-called F-ITERATION or V-ITERATION based on the properties of the trial step $s_k := n_k + t_k$. In an F-ITERATION, the main focus is on decreasing the objective function value while the main focus in an V-ITERATION is on decreasing the infeasibility measure, v. After finding the trial step s_k, a set of simple conditions is checked to find the eligibility of an F-ITERATION. In a typical trust funnel algorithm, these conditions are

$$t_k \neq 0 \tag{1.36a}$$

$$v(x_k + n_k + t_k) \le v_k^{\max} \tag{1.36b}$$

$$m_k^f(0) - m_k^f(n_k+t_k) \ge \kappa\left(m_k^f(n_k) - m_k^f(n_k+t_k)\right), \ \text{where } \kappa \in (0,1). \tag{1.36c}$$

If the conditions for performing an F-ITERATION are not satisfied, then a V-ITERATION is performed. In both an F-ITERATION and a V-ITERATION, the acceptance of the trial step and updating of the trust region radius follow by trust region rules. In a V-ITERATION, the quality of the trial step based on the infeasibility measure decrease is explored, then

accordingly, the funnel radius $v_{k+1}^{\max}$ is set. Algorithm 4 shows a version of a trust funnel algorithm [27].

Algorithm 4 Trust Funnel Algorithm

Require: an acceptance constant $\eta \in \mathbb{R}_{++}$ for accepting steps in F-iteration and V-iteration, with $0 < \eta < 1$, some constants: $0 < \kappa < 1$, $\kappa_v > 0$, $\kappa_{fv} > 0$, $0 < \kappa_{cs} \leq 1$, $0 < \kappa_{ns} \leq 1$, $\kappa_{ts1} > 0$, and $0 < \kappa_{ts2} < 1$.

Require: a constant $\theta \in R_{++}$ with $0 < \theta < 1$ which identifies the fraction of trust region radius in computing normal step.

Require: bound constants $\{\underline{\sigma}, \overline{\sigma}\} \subset \mathbb{R}_{++}$ with $0 \leq \underline{\sigma} \leq \overline{\sigma}$;

procedure Trust Funnel Algorithm
  choose $x_0 \in \mathbb{R}^n$, $y_{-1} \in \mathbb{R}_+^m$, $v_0^{\max} \geq \max\{1, v(x_0)\}$, $\delta_0^f \in \mathbb{R}_{++}$, $\delta_0^v \in \mathbb{R}_{++}$
  for $k = 0, 1, 2, \ldots$ **do**
    calculate v_k
    if $c_k \neq 0$ **then**
      compute normal step, n_k, using (1.33);
    else
      set $n_k \leftarrow 0$;
    compute the Lagrangian multiplier, y_k, using (1.34);
    calculate $g_k + H_k n_k + J_k^T y_k$;
    if $c_k = 0$ and $g_k + H_k n_k + J_k^T y_k = 0$ **then**
      terminate.
    compute tangential step, t_k, using (1.35);
    set $s_k \leftarrow n_k + t_k$;
    if (1.36) is satisfied **then** [F-iteration]
      set $v_{k+1}^{\max} \leftarrow v_k^{\max}$
      compute $\rho_k^f := \frac{f(x_k) - f(x_k + s_k)}{m_k^f(0) - m_k^f(s_k)}$;
      if $\rho_k^f \geq \eta$ **then**
        set $x_{k+1} \leftarrow x_k + s_k$, $\delta_{k+1}^v \geq \delta_k^v$, $\delta_{k+1}^f \geq \delta_k^f$
      else
        set $x_{k+1} \leftarrow x_k$, $\delta_{k+1}^v \leftarrow \delta_k^v$, $\delta_{k+1}^f \in (0, \delta_k^f)$
    else [V-iteration]
      set $\delta_{k+1}^f \leftarrow \delta_k^f$
      compute $\rho_k^v := \frac{v(x_k) - v(x_k + s_k)}{m_k^v(0) - m_k^v(s_k)}$;
      if $\rho_k^v \geq \eta$ **then**
        set $x_{k+1} \leftarrow x_k + s_k$, $\delta_{k+1}^v \geq \delta_k^v$, $v_{k+1}^{\max} \in (0, v_k^{\max})$
      else
        set $x_{k+1} \leftarrow x_k$, $\delta_{k+1}^v \in (0, \delta_k^v)$, $v_{k+1}^{\max} \leftarrow v_k^{\max}$

Algorithm 4 is globally convergent [27, 38]. However, to the best of our knowledge, its complexity properties have not been studied in detail. In Chapter 4, we propose a modification of this framework with good worst-case complexity bounds. Our algorithm

borrows many ideas from the method proposed in Chapter 2.

1.2.4.4 The Short-Step ARC (ShS-ARC) algorithm

The success of ARC in approximately solving unconstrained problems within $\mathcal{O}(\epsilon^{-3/2})$ iterations drew researchers' attention to designing algorithms with improved iteration complexity bounds for constrained optimization. Cartis, Gould, and Toint explored the complexity of constrained optimization [18, 19]. Particularly, in [19], they introduce a two-phase method for (1.23), where in both phases ARC was used. They analyze the worst-case iteration complexity bound of the proposed algorithm to locate a relative KKT point and prove that it can even be as good as $\mathcal{O}(\epsilon^{-3/2})$, under some assumptions.

In the Short-Step ARC (ShS-ARC) algorithm (see Algorithm 5), the first phase is used to find a (relatively) feasible point by using ARC to minimize $\frac{1}{2}\|c(x)\|_2^2$. The second phase, on the other hand, is trying to decrease the objective function within the relative feasible region through a so-called "target following" strategy. During the second phase, a target value t for the objective function is introduced. Then, ARC is used to reduce the $\frac{1}{2}\|r(x,t)\|_2^2$, where

$$r(x,t) := \begin{pmatrix} c(x) \\ f(x) - t \end{pmatrix}. \tag{1.37}$$

The target value t is carefully chosen and iteratively decreased until the termination condition is satisfied. The termination conditions involve a function called "scaled gradient" as a measure of optimality such that

$$g_r(x,t) := \begin{cases} \frac{\nabla c(x)^T c(x) + (f(x)-t)\nabla f(x)}{\|r(x,t)\|_2} & \text{whenever } r(x,t) \neq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{1.38}$$

Phase 1 terminates either with an approximately feasible point x_0 such that $\|c(x_0)\|_2 \leq \epsilon_p$ or an approximate infeasible first-order stationary point of $\|c(x)\|_2$. The number of Phase 1 iterations to generate a point with $\|c(x_1)\|_2 \leq \epsilon_p$ or norm of its gradient, $\frac{\|\nabla c(x_1)^T c(x_1)\|_2}{\|c(x_1)\|_2} \leq \epsilon_d$ is at most $\mathcal{O}(\epsilon^{-3/2})$ with $\epsilon = \min\{\epsilon_p, \epsilon_d\}$. In Phase 2, on the other hand, the algorithm starts from an approximately feasible point x_1 generated by Phase 1, keeps the relative feasibility such that for all iteration $k \geq 0$, $\|c(x_k)\|_2 \leq \epsilon_p$ while trying to decrease the distance of the function value with a monotonically decreasing sequence

Algorithm 5 The Short-Step ARC (ShS-ARC) algorithm

Require: a starting point x_0,
Require: initial regularization parameters σ_0 and σ_1 and a minimal one $\sigma_{\min}$ such that $\min\{\sigma_0, \sigma_1\} \geq \sigma_{\min} > 0$,
Require: algorithmic parameters $\gamma_2 \geq \gamma_1 > 1$ and $1 > \eta_2 \geq \eta_1 > 0$,
Require: the tolerances $\epsilon_p \in (0,1)$ and $\epsilon_d \in (0,1)$.

procedure PHASE 1
  starting from x_0, apply ARC to minimize $\frac{1}{2}\|c(x)\|_2^2$ until a point x_1 is found such that

$$\|c(x_1)\|_2 \leq \epsilon_p \quad \text{or} \quad \frac{\|\nabla c(x_1)^T c(x_1)\|_2}{\|c(x_1)\|_2} \leq \epsilon_d. \tag{1.39}$$

  if $\|c(x_1)\|_2 > \epsilon_p$ **then**
    terminate. (locally infeasible)

procedure PHASE 2
  set $t_1 \leftarrow f(x_1) - \sqrt{\epsilon_p^2 - \|c(x_1)\|_2^2}$ and $k \leftarrow 1$
  for $k = 1, 2, \ldots$ **do**
    starting from x_k, apply one iteration of ARC to approximately minimize

$$\frac{1}{2}\left\|\left(c(x_k)^T, f(x_k) - t_k\right)\right\|_2$$

    if $\rho_k^C \geq \eta_1$ **then**
      if $\|g_r(x_{k+1}, t_k)\|_2 \leq \epsilon_d$ and $r(x_{k+1}, t_k) \neq 0$ **then**
        terminate.
      else
        set

$$t_{k+1} = f(x_{k+1}) - \sqrt{\|r(x_k, t_k)\|_2^2 - \|r(x_{k+1}, t_k)\|_2^2 + \left(f(x_{k+1} - t_k\right)^2}.$$

    else
      set $t_{k+1} \leftarrow t_k$

$\{t_k\}$, such that $|f(x_k) - t_k| \leq \epsilon_p$. All these are guaranteed by the definition of t_1 in Step 6 and the updating of t_k in Steps 13 and 15. Illustration of Phases 1 and 2 of ShS-ARC can be seen in Figure 1.3.

When Phase 2 terminates, we are either at an approximate stationary point of $\|c(x)\|_2$ with $f(x) = t$, or at a relative KKT point with $f(x) \neq t$ ([19], Lemma 4.2) such that

$$\frac{\|\nabla c(x)^T y(x,t) + \nabla f(x)\|_2}{\|(y(x,t), 1)\|_2} \leq \epsilon_d, \tag{1.40}$$

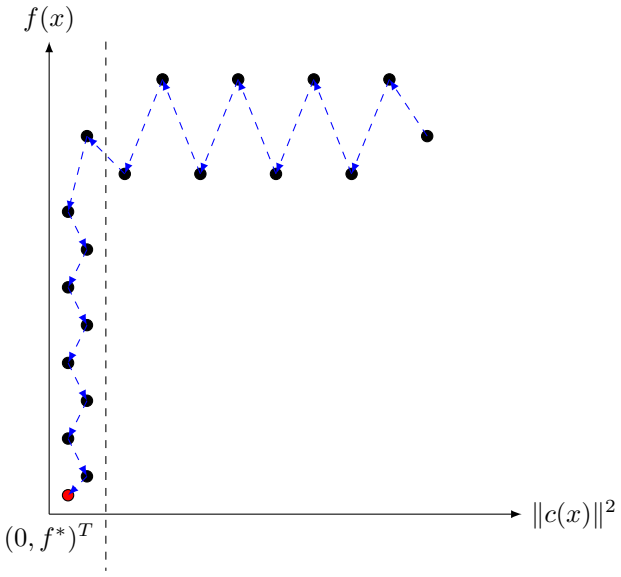


Figure 1.3: Illustration of Phases 1 and 2 of ShS-ARC

where

$$y(x,t) := \frac{c(x)}{|f(x)-t|}.$$

In [19], the authors have justified the condition (1.40) by a perturbation argument: Consider $x = x^* + \delta_x$ and $y = y^* + \delta_y$ where (x^*, y^*) is the primal-dual pair satisfying KKT conditions. Then, a first-order Taylor's expansion of $\nabla c(x^*)^T y^* + \nabla f(x^*)$ to estimate its value at the perturbed point (x, y) will give us the estimation $\left(\nabla^2 f(x^*) + \sum_{i=1}^m y_i^* \nabla^2 c_i(x^*)\right) \delta_x + \nabla c(x^*)^T \delta_y$. The presence of dual variable y^* in this estimation justifies that the magnitude of the dual variable should not be ignored in relative KKT point condition as in (1.40).

In the next theorem, the iteration complexity bound for Algorithm 5 and the required assumptions are represented from [19].

Theorem 2. *Assume that:*

- *The function c is twice continuously differentiable on $\mathbb{R}^n$ and f is twice continuously differentiable in a sufficiently large open set containing $\mathcal{C}_1 := \{x \in \mathbb{R}^n \mid \|c(x)\|_2 \leq \kappa_c\}$, where $\kappa_c > \epsilon_p$.*
- *The Jacobian $\nabla c(x)$, the components $c_i(x)$, and $\nabla^2 c_i(x)$ for $i \in \{1, 2, \ldots, m\}$ are*

globally Lipschitz continuous on the path of all Phase 1 and Phase 2 iterates and trial points.

- *$f(x)$, $\nabla f(x)$, and $\nabla^2 f(x)$ are globally Lipschitz continuous on the path of all Phase 2 iterates and trial points.*
- *The objective function $f(x)$ is bounded above and below in $\mathcal{C}_1$.*

In addition, consider $\epsilon_d \leq \epsilon_p^{1/3}$. Furthermore, assume that the Hessian approximates used in ARC *subproblems to minimize $\frac{1}{2}\|c(x)\|_2^2$ in Phase 1 and $\frac{1}{2}\|r(x,t_k)\|_2^2$ in Phase 2 are relatively accurate. Then, Algorithm 5 generates an iterate x_k satisfying either a relative KKT condition for (1.23) with*

$$\|c(x_k)\|_2 \leq \epsilon_p \quad \text{and} \quad \frac{\|\nabla c(x_k)^T y_k + \nabla f(x_k)\|_2}{\|(y_k, 1)\|_2} \leq \epsilon_d$$

for some $y_k \in \mathbb{R}^m$, or an approximate first-order stationary point for $\|c(x)\|_2$ with

$$\frac{\|\nabla c(x_k)^T c(x_k)\|_2}{\|c(x_k)\|_2} \leq \epsilon_d$$

in at most $\mathcal{O}(\epsilon_d^{-3/2}\epsilon_p^{-1/2})$ evaluations of c and f and their derivatives. Therefore, if one chooses $\epsilon := \epsilon_d = \epsilon_p^{2/3}$, then the complexity bounds will be $\mathcal{O}(\epsilon^{-3/2})$, the same as the unconstrained case when applying ARC *to solve.*

Despite the worst-case iteration complexity bound of ShS-ARC, its performance in practice is yet to be explored. In addition, the relatively poor performance of ARC in unconstrained problems in compared to trust region algorithm raises the question of whether it is possible to design a trust region based algorithm for constrained problem (1.23) with the same complexity bound of ShS-ARC. In Chapter 4, an algorithm is proposed which uses the trust funnel framework to guarantee convergence while using TRACE ideas (introduced in Chapter 2) to prove the worst-case iteration complexity bound. We believe that the resulting algorithm will be more practical because, in contrast to the ShS-ARC, our algorithm always uses constraint and objective functions information in every iteration.

Chapter 2

A Trust Region Algorithm with a Worst-Case Iteration Complexity of $\mathcal{O}(\epsilon^{-3/2})$ for Nonconvex Optimization

The purpose of this chapter[1] is to propose and analyze a trust region method that ensures similar global and fast local convergence guarantees as a traditional trust region method and the ARC algorithm, but also attains, under comparable assumptions, the same worst-case iteration complexity bounds proved to hold for ARC. In particular, we show that our algorithm, entitled TRACE, ensures that the stationarity tolerance (1.2) is met after at most $\mathcal{O}(\epsilon^{-3/2})$ iterations, and that a related tolerance for second-order stationarity is met after at most $\mathcal{O}(\epsilon^{-3})$ iterations. (Both of these bounds have the same order for the ARC algorithm.) We show that these properties can be realized within the context of a trust region strategy by employing (i) modified step acceptance criteria and (ii) a novel updating mechanism for the trust region radius. Indeed, our acceptance and updating mechanisms represent significant departures from those employed in a traditional approach; e.g., there are situations in which our algorithm may *reject* a step and *expand* the trust region, and there are situations in which our algorithm sets a subsequent trust region radius *implicitly* via a *quadratic* regularization strategy. In ways that will be revealed in our discussion

[1] A paper containing the original material of this chapter was published in 2017. Please refer to [28].

and analysis, the effect of these changes is that the accepted steps in our algorithm have properties that are similar to those that are critical for ensuring the worst-case complexity bounds for the ARC algorithm. That being said, our algorithm allows for the computation and acceptance of steps that are not regularized (i.e., Newton steps), which is worthwhile to note in contrast to ARC in which the cubic regularization strategy is never "off" [14, 15].

For simplicity in our description and to highlight the salient features of it in our analysis, our algorithm states that second-order derivatives be used and that each trial step be computed as a globally optimal solution of a trust region subproblem. Admittedly, these requirements are impractical in some large scale settings, which is why algorithm variants that only require approximate second-order derivative information and/or subproblem solutions that are only optimal with respect to Krylov subspaces are common in the context of trust region methods; e.g., see [23]. We expect that such variations of our algorithm can be designed that maintain our global convergence guarantees and—with sufficiently accurate second-order derivative approximations and subproblem solutions—our worst-case complexity bounds and local convergence guarantees. (Indeed, in [14, 15], variants of the ARC algorithm are discussed that are suitable for large-scale problems.)

This chapter is organized as follows. In §2.1, we propose our trust region algorithm, highlighting the features that distinguish it from a traditional trust region approach as well as those that allow it to obtain improved complexity properties. We prove convergence guarantees for the algorithm in §2.2, illustrating that it converges globally from remote starting points (see §2.2.1), reduces the norm of the gradient of the objective below a prescribed $\epsilon \in (0, \infty)$ after at most $\mathcal{O}(\epsilon^{-3/2})$ iterations (see §2.2.2), yields an approximate second-order stationary point after at most $\mathcal{O}(\epsilon^{-3})$ iterations (see §2.2.3), and attains a Q-quadratic rate of local convergence under standard assumptions for a trust region methods (see §2.2.4). Finally, the result of our preliminary numerical experiments is presented in §2.3.

Notation. Given an iterate x_k in an algorithm for solving (1.1), henceforth we define $f_k := f(x_k)$, $g_k := g(x_k) := \nabla f(x_k)$, and $H_k := H(x_k) := \nabla^2 f(x_k)$, where $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$ has already been defined as the gradient function of f and $\nabla^2 f : \mathbb{R}^n \to \mathbb{R}^{n \times n}$ is its Hessian function. Similarly, we apply a subscript to other algorithmic quantities whose definition depends on the iteration number k. We use $\mathbb{R}_+$ to denote the set of nonnegative scalars, $\mathbb{R}_{++}$ to denote the set of positive scalars, $\mathbb{N}_+$ to denote the set of nonnegative

integers, and $\mathbb{N}_{++}$ to denote the set of positive integers. Given a real symmetric matrix A, we write $A \succeq 0$ (respectively, $A \succ 0$) to indicate that A is positive semidefinite (respectively, positive definite). Given a pair of scalars $(a, b) \in \mathbb{R} \times \mathbb{R}$, we write $a \perp b$ to indicate that $ab = 0$. Similarly, given such a pair, we denote their maximum as $\max\{a, b\}$, their minimum as $\min\{a, b\}$, and, when $(a, b) \in \mathbb{R}_{++} \times \mathbb{R}_{++}$, we use the convention that $\min\{a, b/0\} = a$. Finally, given a discrete set $\mathcal{S}$, we denote its cardinality by $|\mathcal{S}|$.

Citations. Our analysis makes extensive use of Taylor's Theorem, the Mean Value Theorem, the Cauchy-Schwarz inequality, and the Triangle Inequality. However, for brevity, we do not cite these tools in each instance in which they are used.

2.1 Algorithm Description

In this section, we formally propose our algorithm. Given an initial point $x_0 \in \mathbb{R}^n$, our algorithm follows the typical trust region strategy of computing a sequence of iterates $\{x_k\} \subset \mathbb{R}^n$, where at x_k we compute a trial step by minimizing a local quadratic model of f at x_k within a trust region described by a positive trust region radius. Distinguishing features of our algorithm are a set of step acceptance criteria and an update mechanism for the trust region radius that are different than those employed in a traditional trust region algorithm.

We make the following assumption about f throughout this chapter.

Assumption 3. *The objective function f is twice continuously differentiable on $\mathbb{R}^n$.*

At an iterate x_k, we define $q_k : \mathbb{R}^n \to \mathbb{R}$ as a second-order Taylor series approximation of f about x_k, i.e., we let q_k be defined by

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

We also define, as dependent on a given iterate x_k and trust region radius $\delta_k > 0$, the trust region subproblem

$$\mathcal{Q}_k : \quad \min_{s \in \mathbb{R}^n} \; q_k(s) \;\; \text{subject to} \;\; \|s\|_2 \leq \delta_k.$$

As in a traditional trust region algorithm, the primary computational expense in iteration

k of our algorithm is incurred in solving a trust region subproblem of this form. Given (x_k, δ_k), it is well known [23, Corollary 7.2.2] that a globally optimal solution of $\mathcal{Q}_k$ exists and is given by any vector s_k to which there corresponds a dual variable, call it λ_k, such that the following conditions are satisfied:

$$g_k + (H_k + \lambda_k I)s_k = 0 \tag{2.1a}$$

$$(H_k + \lambda_k I) \succeq 0 \tag{2.1b}$$

$$0 \leq \lambda_k \perp (\delta_k - \|s_k\|_2) \geq 0. \tag{2.1c}$$

(Strictly speaking, since the global minimizer of $\mathcal{Q}_k$ may not be unique, the vector s_k is not uniquely determined in this manner; for our purposes, however, it suffices to let s_k denote any global minimizer of $\mathcal{Q}_k$.) For future reference, we also remark that, for a given scalar $\lambda \geq 0$ that is strictly larger than the negative of the leftmost eigenvalue of H_k, (2.1) implies that the solution $s \in \mathbb{R}^n$ of the subproblem

$$\mathcal{Q}_k(\lambda): \quad \min_{s \in \mathbb{R}^n} \; f_k + g_k^T s + \tfrac{1}{2} s^T (H_k + \lambda I) s,$$

or, equivalently, the solution of the linear system

$$g_k + (H_k + \lambda I)s = 0, \tag{2.2}$$

corresponds to a globally optimal solution of a trust region subproblem with trust region radius $\delta = \|s\|_2$. That is, by perturbing $\mathcal{Q}_k$ through the addition of a *quadratic* regularization term with coefficient λ to obtain $\mathcal{Q}_k(\lambda)$, we obtain the solution of a trust region subproblem for an *implicitly* defined trust region radius.

2.1.1 Motivation for TRACE

We motivate the design of TRACE by first recalling the procedures of a traditional trust region algorithm. In such a method, after the trial step s_k is computed via $\mathcal{Q}_k$ according to the trust region radius $\delta_k > 0$, the remainder of iteration k involves one of two possible outcomes: either the trial step is accepted—in which case the next iterate is set as $x_{k+1} \leftarrow x_k + s_k$ and one may choose $\delta_{k+1} \geq \delta_k$—or rejected—in which case the next iterate is set as $x_{k+1} \leftarrow x_k$ and one must choose $\delta_{k+1} < \delta_k$. The typical step acceptance criterion

employed to determine which of these outcomes is to be realized involves the ratio of the actual-to-predicted reduction in the objective yielded by the trial step, i.e.,

$$\frac{f_k - f(x_k + s_k)}{f_k - q_k(s_k)}. \tag{2.3}$$

If this ratio is larger than a prescribed constant in $(0, 1)$, then the step is accepted, and otherwise it is rejected. (In the latter case, the contraction of the trust region radius may continue iteratively until an acceptable step is computed.) Overall, in such an approach, each iteration can be characterized as one in which either a trial step is accepted or the trust region is contracted. Furthermore, the typical strategy of updating the trust region radius involves multiplying the previous radius by a constant factor; such an approach implies, e.g., that if the trust region radius is being reduced, then it is done so at a *linear* rate.

When it comes to deriving worst-case iteration complexity bounds for a traditional trust region method, an acceptance criterion based on the ratio (2.3) may yield accepted steps that do not yield the level of decrease in the objective that is required to yield complexity properties that are competitive with those of ARC. Moreover, such accepted steps may not be sufficiently large in norm in order to guarantee a sufficient decrease in the norm of the gradient of the objective, which is critical for driving this quantity to zero at a fast rate. Overall, by observing the analysis for ARC in [15], one finds that the algorithm computes a positive sequence of cubic regularization coefficients $\{\sigma_k\} \subset [\sigma_{min}, \sigma_{max}]$ for some user-specified $\sigma_{min} \in (0, \infty)$ and unknown problem-dependent constant $\sigma_{max} \in [\sigma_{\min}, \infty)$ that satisfy the following two critical properties for any accepted step:

$$f_k - f_{k+1} \geq c_1 \sigma_k \|s_k\|_2^3 \quad \text{and} \quad \|s_k\|_2 \geq \left(\frac{c_2}{\sigma_{max} + c_3}\right)^{1/2} \|g_{k+1}\|_2^{1/2}, \tag{2.4}$$

where $\{c_1, c_2, c_3\} \subset \mathbb{R}_{++}$ are constants that are independent of $\epsilon \in (0, \infty)$. (These inequalities can be seen in [15, Lemma 4.2, Lemma 5.1, Lemma 5.2]. By combining the inequalities, one obtains the power of $3/2$ that appears in the complexity bound for ARC in terms of finding an approximate first-order stationary point. Moreover, the first inequality leads to the power of 3 that appears in the complexity bound for attaining approximate second-order stationarity.) In a traditional trust region algorithm, inequalities of this type are not guaranteed [16].

2.1.2 A detailed description of TRACE

Motivated by the discussion in the previous subsection, we designed step acceptance criteria and an updating mechanism for the trust region radius in TRACE that guarantee that all accepted steps possess properties that are similar to those in (2.4). We ensure these critical properties by modifying a traditional trust region framework in three key ways, as described in the following bullets. (A formal statement of TRACE is presented as Algorithm 6 on page 39 and should be referenced as needed for additional context while reading the remainder of this subsection.)

- Our simplest modification is that we measure sufficient decrease in the objective function by observing a ratio inspired by the former inequality in (2.4). Specifically, for all $k \in \mathbb{N}_+$, we define (as opposed to (2.3)) the ratio

$$\rho_k := \frac{f_k - f(x_k + s_k)}{\|s_k\|_2^3}. \tag{2.5}$$

  For a prescribed scalar $\eta \in (0, 1)$, a trial step may only be considered acceptable if $\rho_k \geq \eta$; however, not all steps satisfying this condition will be accepted. (We provide further motivation for this condition in §2.1.3.)

- In a traditional trust region algorithm, it is entirely possible that—due to a small magnitude of the trust region radius—a trial step may yield a relatively large decrease in the objective (e.g., $\rho_k \geq \eta$), but not satisfy a condition such as the latter inequality in (2.4). That is, the step may yield a relatively large objective reduction, but may not be sufficiently large in norm, from which it may follow that a sequence of such steps may not drive the gradient of the objective to zero at a fast rate. In TRACE, we avoid such a possibility by incorporating a novel strategy of potentially *rejecting* a trial step in conjunction with an *expansion* of the trust region. The conditions under which we may make such a decision relate to the magnitude of the dual variable λ_k for the trust region constraint corresponding to the subproblem solution s_k.

- Our third modification relates to the manner in which the trust region radius is decreased after a trial step is rejected. As previously mentioned, in a traditional strategy, a contraction of the trust region involves a *linear* rate of decrease of the

radius. In certain cases, such a rate of decrease may have a detrimental effect on the worst-case iteration complexity of the method; specifically, after a single decrease, the radius may jump from a value for which $\rho_k < \eta$ to a value for which the norm of the resulting trial step is not sufficiently large in norm (as described in the second bullet above). In our algorithm, we confront this issue by designing a trust region updating mechanism that may, in certain cases, lead to a *sublinear* rate of decrease in the trust region radius. In particular, in any situation in which the trust region radius is to decrease, we compare the radius that would be obtained via a traditional updating scheme to the norm of the trial step obtained from $\mathcal{Q}_k(\lambda)$ for a carefully chosen $\lambda > 0$. If the norm of the step resulting from this procedure falls into a suitable range, then we employ it as the trust region radius in the subsequent iteration, as opposed to updating the radius in a more traditional manner. (We remark that if a sequence of consecutive trial steps are rejected in the ARC algorithm, then one finds that the norms of the trial steps converge to zero sublinearly. We believe this feature is critical in its ability to provide an improved worst-case complexity bound as compared to a traditional trust region method.)

Given this overview, we now state that our step acceptance criteria and trust region updating mechanism make use of three sequences in addition to the standard sequences $\{x_k\}$, $\{\delta_k\}$, and $\{\rho_k\}$. Our first such sequence is the sequence of dual variables $\{\lambda_k\}$. We observe these values to avoid the acceptance of a step that yields a large decrease in the objective function (relative to $\|s_k\|_2$), but for which $\|s_k\|_2$ is deemed too small. (Recall the second bullet above.) Note that, as a dual variable for the trust region constraint, a large value for λ_k as compared to $\|s_k\|_2$ suggests that an even *larger* reduction in the objective function may be achieved by *expanding* the trust region. In certain cases, our algorithm deems such an increase to be necessary in order to compute an acceptable step with desirable properties.

Our second auxiliary sequence is a positive parameter sequence that is set dynamically within the algorithm. This sequence, which we denote as $\{\sigma_k\}$, plays a similar *theoretical* role as the sequence of cubic regularization coefficients in the ARC algorithm. In particular, we use this sequence to estimate an upper bound for the ratio $\lambda_k/\|s_k\|_2$ that the algorithm should allow for an acceptable step. When, during iteration $k \in \mathbb{N}_+$, a pair (s_k, λ_k) is computed that both yields a sufficiently large reduction in the objective function (relative

to $\|s_k\|_2$) and yields $\lambda_k/\|s_k\|_2 \leq \sigma_k$, then we set $\sigma_{k+1} \leftarrow \sigma_k$. Indeed, we only potentially set $\sigma_{k+1} > \sigma_k$ when there is reason to believe that such an increase is necessary in order to accept a step yielding a sufficiently large reduction in the objective function. (It is worthwhile to note that, despite the similar *theoretical* role of our sequence $\{\sigma_k\}$ vis-à-vis the sequence of cubic regularization coefficients in the ARC algorithm, our sequence plays a decidedly different *practical* role. In particular, no element of this sequence is involved in the definition of the trust region subproblem $\mathcal{Q}_k$; the sequence $\{\sigma_k\}$ only appears in our step acceptance criteria.)

The third auxiliary sequence employed in our algorithm, denoted by $\{\Delta_k\}$, is a monotonically nondecreasing sequence of upper bounds for the trust region radii. TRACE sets $\Delta_{k+1} > \Delta_k$ whenever s_k is accepted and $\|s_k\|_2$ is sufficiently large compared to Δ_k. The role of this sequence is to cap the magnitude of the trust region radius when it is increased, which is needed in our analysis.

Since our method involves *contractions* of the trust region and a novel *expansion* procedure, we have chosen the name TRACE as an acronym for *Trust Region Algorithm with Contraction and Expansion*. In many ways, our approach follows the same strategy of a traditional trust region algorithm, except as far as the CONTRACT subroutine and expansion of the trust region in Step 17 are concerned. To motivate these procedures, we provide the following remarks.

- Any call to the CONTRACT subroutine is followed by Step 19 in which a trust region subproblem is solved. However, in many cases, the solution of this trust region subproblem has already been computed within the CONTRACT subroutine. Certainly, an efficient implementation of TRACE would avoid re-solving a trust region subproblem in such cases; we have merely written the algorithm in this manner to illustrate its structure as a trust region method and so that our analysis may be more simply presented. We also note that while the CONTRACT subroutine involves the solution of a linear system involving a symmetric positive definite matrix, this should not be considered as an additional computational expense in our method. Indeed, the solutions of such systems are required in certain implementations of second-order trust region algorithms. In Appendix A, we address this issue in further detail, showing that our algorithm can be implemented in such a way that each iteration is *at most* as expensive as that of a traditional trust region algorithm or ARC.

- The procedure in CONTRACT—specifically, the values for λ in Steps 25 and 33, as well as the order of operations and conditional statements for setting δ_{k+1}—has been designed in such a way that, corresponding to the newly computed trust region radius δ_{k+1}, the primal-dual solution (s_{k+1}, λ_{k+1}) of $\mathcal{Q}_{k+1}$ satisfies

$$\lambda_{k+1} \geq \underline{\sigma} \|s_{k+1}\|_2.$$

  Moreover, under the assumptions used in our complexity analysis, we show that the procedure in CONTRACT ensures that the sequence $\{\lambda_{k+1}/\|s_{k+1}\|_2\}$—and, consequently, the sequence $\{\sigma_k\}$—will be bounded above. (See Lemma 15.)

- Perhaps the most intriguing aspect of our algorithm is the calculation stated in Step 30; indeed, in this step, the algorithm requests the solution of a trust region subproblem for which neither the trust region radius nor a quadratic regularization coefficient has been explicitly specified. We claim, however, that this calculation can actually be *less* expensive than the solution of an explicit subproblem that would arise in either a traditional trust region algorithm or ARC. Indeed, one may view this step as requesting the solution of an ARC subproblem for *any* cubic regularization coefficient in the range $[\underline{\sigma}, \overline{\sigma}]$. We discuss a practical implementation of this step in Appendix A.

- The update for the trust region radius in Step 17 is designed so $\delta_{k+1} > \delta_k$. In fact, we prove in our analysis that, as a result of such an expansion, the subsequent iteration will either involve an accepted step or a contraction of the trust region, and that another expansion cannot occur until another step has been accepted. (See Lemma 6.) Overall, an expansion of the trust region aids in avoiding steps that are too small in norm, and our particular formula for an expansion guarantees that at most one expansion can occur between accepted steps, which is critical in our complexity analysis.

2.1.3 Further discussion on step acceptance in TRACE

We close this section with further motivation for the acceptance condition $\rho_k \geq \eta$ (which clearly emulates the former condition in (2.4)). As is known in the theory of trust region

Algorithm 6 Trust Region Algorithm with Contraction and Expansion (TRACE)

Require: an acceptance constant $\eta \in \mathbb{R}_{++}$ with $0 < \eta < 1$
Require: update constants $\{\gamma_{\text{C}}, \gamma_{\text{E}}, \gamma_\lambda\} \subset \mathbb{R}_{++}$ with $0 < \gamma_{\text{C}} < 1 < \gamma_{\text{E}}$ and $\gamma_\lambda > 1$
Require: ratio bound constants $\{\underline{\sigma}, \overline{\sigma}\} \subset \mathbb{R}_{++}$ with $0 < \underline{\sigma} \leq \overline{\sigma}$

procedure TRACE
  choose $x_0 \in \mathbb{R}^n$, $\{\delta_0, \Delta_0\} \subset \mathbb{R}_{++}$ with $\delta_0 \leq \Delta_0$, and $\sigma_0 \in \mathbb{R}_{++}$ with $\sigma_0 \geq \underline{\sigma}$
  compute (s_0, λ_0) by solving $\mathcal{Q}_0$, then set ρ_0 as in (2.5)
  for $k = 0, 1, 2, \ldots$ **do**
    if $\rho_k \geq \eta$ and either $\lambda_k \leq \sigma_k \|s_k\|_2$ or $\|s_k\|_2 = \Delta_k$ **then** [accept step]
      set $x_{k+1} \leftarrow x_k + s_k$
      set $\Delta_{k+1} \leftarrow \max\{\Delta_k, \gamma_{\text{E}} \|s_k\|_2\}$
      set $\delta_{k+1} \leftarrow \min\{\Delta_{k+1}, \max\{\delta_k, \gamma_{\text{E}} \|s_k\|_2\}\}$
      set $\sigma_{k+1} \leftarrow \max\{\sigma_k, \lambda_k / \|s_k\|_2\}$
    else if $\rho_k < \eta$ **then** [contract trust region]
      set $x_{k+1} \leftarrow x_k$
      set $\Delta_{k+1} \leftarrow \Delta_k$
      set $\delta_{k+1} \leftarrow \text{CONTRACT}(x_k, \delta_k, \sigma_k, s_k, \lambda_k)$
    else (i.e., **if** $\rho_k \geq \eta$, $\lambda_k > \sigma_k \|s_k\|_2$, and $\|s_k\|_2 < \Delta_k$) [expand trust region]
      set $x_{k+1} \leftarrow x_k$
      set $\Delta_{k+1} \leftarrow \Delta_k$
      set $\delta_{k+1} \leftarrow \min\{\Delta_{k+1}, \lambda_k / \sigma_k\}$
      set $\sigma_{k+1} \leftarrow \sigma_k$
    compute (s_{k+1}, λ_{k+1}) by solving $\mathcal{Q}_{k+1}$, then set ρ_{k+1} as in (2.5)
    if $\rho_k < \eta$ **then**
      set $\sigma_{k+1} \leftarrow \max\{\sigma_k, \lambda_{k+1} / \|s_{k+1}\|_2\}$

procedure CONTRACT$(x_k, \delta_k, \sigma_k, s_k, \lambda_k)$
  if $\lambda_k < \underline{\sigma} \|s_k\|_2$ **then**
    set $\hat{\lambda} \leftarrow \lambda_k + (\underline{\sigma} \|g_k\|_2)^{1/2}$
    set $\lambda \leftarrow \hat{\lambda}$
    set s as the solution of $\mathcal{Q}_k(\lambda)$ [$\underline{\sigma} \leq \lambda / \|s\|_2$ holds; see Lemma 14]
    if $\lambda / \|s\|_2 \leq \overline{\sigma}$ **then**
      return $\delta_{k+1} \leftarrow \|s\|_2$
    else
      compute $\lambda \in (\lambda_k, \hat{\lambda})$ so the solution s of $\mathcal{Q}_k(\lambda)$ yields $\underline{\sigma} \leq \lambda / \|s\|_2 \leq \overline{\sigma}$
      return $\delta_{k+1} \leftarrow \|s\|_2$
  else (i.e., **if** $\lambda_k \geq \underline{\sigma} \|s_k\|_2$)
    set $\lambda \leftarrow \gamma_\lambda \lambda_k$
    set s as the solution of $\mathcal{Q}_k(\lambda)$
    if $\|s\|_2 \geq \gamma_{\text{C}} \|s_k\|_2$ **then**
      return $\delta_{k+1} \leftarrow \|s\|_2$
    else
      return $\delta_{k+1} \leftarrow \gamma_{\text{C}} \|s_k\|_2$

algorithms (and can be seen in the proof of Lemma 24 in §2.2.4), one can show in a neighborhood of a strict local minimizer x_* with $H(x_*) \succ 0$ that the decrease of the objective function yielded by a Newton step is expected to be $\xi_* \|s_k\|_2^2$ where ξ_* is a constant related to the condition number and norm of the inverse Hessian of f at x_*. Correspondingly, a reasonable step acceptance criterion is one that requires objective function decrease of this order. One way in which to design such a criterion is to employ the ratio (2.3) in the traditional manner of a trust region algorithm. Alternatively, one may be more explicit in such a ratio and replace the denominator with a quantity such as $\xi \|s_k\|_2^2$ for some $\xi > 0$. However, an issue with this approach is that, unless one knows *a priori* how to choose $\xi \in (0, \xi_*]$, such a step acceptance criterion may reject Newton steps in a neighborhood of x_*, which may impede fast local convergence. Our approach is to require that any accepted step yields a reduction in the objective function that is proportional to $\|s_k\|_2^3$, which is reasonable when $\{s_k\} \to 0$. These observations offer intuitive evidence for our claim (proved in §2.2.4) that, if $\{x_k\}$ converges to a local minimizer of f satisfying certain properties, then our algorithm will compute and accept Newton steps asymptotically.

On the other hand, one needs to consider the implications of our step acceptance criteria when not in a neighborhood of a strict local minimizer. Indeed, while perhaps ensuring the acceptance of Newton steps that are (relatively) small in norm, such a criterion may reject those that are large in norm. We believe that the appropriateness of the criterion can still be justified given that, when far from a stationary point, one cannot always guarantee the acceptance (under *any* reasonable criteria) of full Newton steps. In fact, often, full Newton steps may not even be feasible for the trust region subproblem. Therefore, while our step acceptance criterion may reject large Newton steps, the added benefits—as shown in this chapter—are improved worst-case iteration complexity bounds.

As a side note, we remark that the ARC algorithm only achieves its worst-case complexity bounds by imposing a uniform lower bound on the cubic regularization coefficient, which implies that the ARC algorithm never computes Newton steps, even in a neighborhood of a strict local minimizer. (Nonetheless, quadratic convergence of ARC has still been established.) Our algorithm clearly differs in this regard as our step acceptance criteria may allow a full Newton step as long as it yields a sufficiently large reduction in the objective function.

2.2 Convergence and Worst-Case Iteration Complexity Analyses

In this section, we analyze the convergence and worst-case iteration complexity properties of TRACE. In addition to Assumption 3, we make a few additional assumptions related to the objective function and the sequence of computed iterates. These assumptions will be introduced at the beginning of each subsection in this section, as they are needed. For brevity, we do not remind the reader of each standing assumption that is needed in the statement of each lemma, but, for clarity, we do state the assumptions that are needed in each theorem.

Throughout the analysis of our algorithm, we distinguish between different types of iterations by partitioning the set of iteration numbers into what we refer to as the sets of accepted ($\mathcal{A}$), contraction ($\mathcal{C}$), and expansion ($\mathcal{E}$) steps:

$$\begin{aligned}
\mathcal{A} &:= \{k \in \mathbb{N}_+ : \rho_k \geq \eta \text{ and either } \lambda_k \leq \sigma_k \|s_k\|_2 \text{ or } \|s_k\|_2 = \Delta_k\}, \\
\mathcal{C} &:= \{k \in \mathbb{N}_+ : \rho_k < \eta\}, \text{ and} \\
\mathcal{E} &:= \{k \in \mathbb{N}_+ : k \notin \mathcal{A} \cup \mathcal{C}\}.
\end{aligned}$$

We also partition the set of accepted steps into two disjoint subsets:

$$\mathcal{A}_\Delta := \{k \in \mathcal{A} : \|s_k\|_2 = \Delta_k\} \quad \text{and} \quad \mathcal{A}_\sigma := \{k \in \mathcal{A} : k \notin \mathcal{A}_\Delta\}.$$

2.2.1 Global convergence to first-order stationarity

Our goal in this subsection is to prove that the sequence of objective function gradients vanishes. As such, and since a practical implementation of TRACE may terminate if the norm of a gradient of the objective is below a prescribed positive threshold, we assume without loss of generality that, at each iterate, the corresponding gradient is nonzero. Throughout this subsection, in addition to Assumption 3, we make the following assumption that is standard for global convergence theory for a trust region method.

Assumption 4. *The objective function f is bounded below on $\mathbb{R}^n$ by a scalar constant $f_{min} \in \mathbb{R}$ and the gradient function g is Lipschitz continuous with a scalar Lipschitz constant $g_{Lip} > 0$ in an open convex set containing the sequences $\{x_k\}$ and $\{x_k + s_k\}$. Furthermore, the gradient sequence $\{g_k\}$ has $g_k \neq 0$ for all $k \in \mathbb{N}_+$ and is bounded in that*

there exists a scalar constant $g_{max} \in \mathbb{R}_{++}$ *such that* $\|g_k\|_2 \leq g_{max}$ *for all* $k \in \mathbb{N}_+$. *It follows from Assumption 3, Lipschitz continuity of* g, *and [54, Lemma 1.2.2] that the Hessian sequence* $\{H_k\}$ *is bounded in norm in that there exists a scalar constant* $H_{max} \in \mathbb{R}_{++}$ *such that* $\|H_k\|_2 \leq H_{max}$ *for all* $k \in \mathbb{N}_+$.

One of our main goals in this subsection is to prove that the set $\mathcal{A}$ has infinite cardinality. Toward this goal, our first result establishes a generic lower bound on the norm of any subproblem solution computed in Step 3 or 19.

Lemma 1. *For any* $k \in \mathbb{N}_+$, *the trial step* s_k *satisfies*

$$\|s_k\|_2 \geq \min\left\{\delta_k, \frac{\|g_k\|_2}{\|H_k\|_2}\right\} > 0. \tag{2.6}$$

Proof. If $\|H_k\|_2 = 0$, then, by (2.1a), $g_k + \lambda_k s_k = 0$, which, since $g_k \neq 0$, means that $\lambda_k \neq 0$ and $\|s_k\|_2 \neq 0$. In fact, along with (2.1c), we have that $\|s_k\|_2 = \delta_k > 0$, in which case (2.6) follows. Now suppose $\|H_k\|_2 \neq 0$. If $\|s_k\|_2 = \delta_k > 0$, then (2.6) again follows, but otherwise, by (2.1), we have $H_k s_k = -g_k$. This implies $\|s_k\|_2 \geq \|g_k\|_2/\|H_k\|_2$, so that (2.6) again holds. □

We now state useful relationships related to the reduction of the model of the objective function yielded by such subproblem solutions.

Lemma 2. *For any* $k \in \mathbb{N}_+$, *the trial step* s_k *and dual variable* λ_k *satisfy*

$$f_k - q_k(s_k) = \tfrac{1}{2} s_k^T (H_k + \lambda_k I) s_k + \tfrac{1}{2}\lambda_k \|s_k\|_2^2 > 0. \tag{2.7}$$

In addition, for any $k \in \mathbb{N}_+$, *the trial step* s_k *satisfies*

$$f_k - q_k(s_k) \geq \tfrac{1}{2}\|g_k\|_2 \min\left\{\delta_k, \frac{\|g_k\|_2}{\|H_k\|_2}\right\} > 0. \tag{2.8}$$

Proof. The equation in (2.7) follows in a straightforward manner from (2.1a). Furthermore, it follows from Lemma 1 that $s_k \neq 0$. Therefore, if $\lambda_k > 0$, then the strict inequality in (2.7) holds since, by (2.1b), we have $H_k + \lambda_k I \succeq 0$. On the other hand, if $\lambda_k = 0$, then we have from (2.1a) that $H_k s_k = -g_k \neq 0$ and from (2.1b) that $H_k \succeq 0$. Combining these facts, we have $s_k^T H_k s_k > 0$, which shows that the strict inequality in (2.7) holds. As for the latter part of the lemma, we remark that the first inequality in (2.8) is standard in

the theory of trust region algorithms for s_k being a solution of a trust region subproblem that yields a reduction in a quadratic model of the objective function that is at least as large as that yielded by the so-called Cauchy step; e.g., see [23, Theorem 6.3.1] or [57, Theorem 4.4]. The strict inequality in (2.8) then follows since $g_k \neq 0$ and $\delta_k > 0$. □

Our next result reveals that the CONTRACT subroutine is guaranteed to yield a decrease in the trust region radius and nondecrease in the dual variable.

Lemma 3. *For any $k \in \mathbb{N}_+$, if $k \in \mathcal{C}$, then $\delta_{k+1} < \delta_k$ and $\lambda_{k+1} \geq \lambda_k$.*

Proof. Suppose that $k \in \mathcal{C}$, in which case δ_{k+1} is set in Step 13 and λ_{k+1} is set in Step 19. We prove the result by considering the various cases that may occur within the CONTRACT subroutine. If Step 28, 31, or 36 is reached, then $\delta_{k+1} \leftarrow \|s\|_2$ where s solves $\mathcal{Q}_k(\lambda)$ for $\lambda > \lambda_k$. In such cases, it follows by the fact that $\lambda > \lambda_k$ and standard trust region theory on the relationship between subproblem solutions and their corresponding dual variables [23, Chap. 7] that we have

$$\delta_{k+1} \leftarrow \|s\|_2 < \|s_k\|_2 \leq \delta_k \quad \text{and} \quad \lambda_{k+1} = \lambda > \lambda_k.$$

The other possibility is that Step 38 is reached, in which case $\delta_{k+1} \leftarrow \gamma_C \|s_k\|_2 < \delta_k$, from which it follows under the same reasoning that $\lambda_{k+1} \geq \lambda_k$. □

Using the previous lemma, we now prove important relationships between the sequences $\{\delta_k\}$ and $\{\Delta_k\}$ computed in the algorithm.

Lemma 4. *For any $k \in \mathbb{N}_+$, there holds $\delta_k \leq \Delta_k \leq \Delta_{k+1}$.*

Proof. First, that $\delta_k \leq \Delta_k$ for all $k \in \mathbb{N}_+$ follows by induction: the inequality holds for $k = 0$ by the initialization of quantities in Step 2 and, assuming that it holds in iteration $k \in \mathbb{N}_+$, the fact that it holds in iteration $k + 1$ follows from the computations in Steps 7, 8, 12, 13, 16, and 17 and the result of Lemma 3 (i.e., for $k \in \mathcal{C}$, we have $\delta_{k+1} \leq \delta_k \leq \Delta_k = \Delta_{k+1}$). Second, the fact that $\Delta_k \leq \Delta_{k+1}$ for all $k \in \mathbb{N}_+$ follows from the computations in Steps 7, 12, and 16. □

We may now prove the following complement to Lemma 3.

Lemma 5. *For any $k \in \mathbb{N}_+$, if $k \in \mathcal{A} \cup \mathcal{E}$, then $\delta_{k+1} \geq \delta_k$.*

Proof. Suppose that $k \in \mathcal{A}$, in which case δ_{k+1} is set in Step 8. By Step 7 and Lemma 4, it follows that $\Delta_{k+1} \leftarrow \max\{\Delta_k, \gamma_{\mathrm{E}}\|s_k\|_2\} \geq \Delta_k \geq \delta_k$, from which it follows that $\delta_{k+1} \leftarrow \min\{\Delta_{k+1}, \max\{\delta_k, \gamma_{\mathrm{E}}\|s_k\|_2\}\} \geq \delta_k$, as desired. Now suppose that $k \in \mathcal{E}$, in which case δ_{k+1} is set in Step 17. By the conditions indicated in Step 14, we have $\lambda_k > \sigma_k\|s_k\|_2 \geq 0$, from which it follows by (2.1c) that $\|s_k\|_2 = \delta_k$. We then have by Step 16, Lemma 4, and the conditions indicated in Step 14 that $\delta_{k+1} \leftarrow \min\{\Delta_{k+1}, \lambda_k/\sigma_k\} \geq \min\{\delta_k, \|s_k\|_2\} = \delta_k$, as desired. □

We next prove a result that is a simple consequence of the manner in which we update the trust region radius and update the sequences $\{\sigma_k\}$ and $\{\Delta_k\}$.

Lemma 6. *For any $k \in \mathbb{N}_+$, if $k \in \mathcal{C} \cup \mathcal{E}$, then $(k+1) \notin \mathcal{E}$.*

Proof. Observe that if $\lambda_{k+1} = 0$, then the conditions in Steps 5 and 10 ensure that $(k+1) \notin \mathcal{E}$. Thus, by (2.1c), we may proceed under the assumption that the trial step and dual variable in iteration $k+1$ satisfy $\|s_{k+1}\|_2 = \delta_{k+1}$ and $\lambda_{k+1} > 0$.

Suppose that $k \in \mathcal{C}$, which implies that $\rho_k < \eta$, and, in turn, that the algorithm sets (in Step 21) $\sigma_{k+1} \geq \lambda_{k+1}/\|s_{k+1}\|_2$. If $\rho_{k+1} \geq \eta$, then it follows that $(k+1) \in \mathcal{A}$. Otherwise, $\rho_{k+1} < \eta$, which implies that $(k+1) \in \mathcal{C}$.

Now suppose that $k \in \mathcal{E}$, from which it follows that

$$\lambda_k > \sigma_k\|s_k\|_2, \quad \delta_{k+1} \leftarrow \min\{\Delta_k, \lambda_k/\sigma_k\} \quad \text{and} \quad \sigma_{k+1} \leftarrow \sigma_k. \tag{2.9}$$

In particular, $\lambda_k > 0$, which implies by (2.1c) that $\|s_k\|_2 = \delta_k$. Consider two cases.

1. Suppose $\Delta_k \geq \lambda_k/\sigma_k$. It then follows from (2.9) that

$$\delta_{k+1} \leftarrow \lambda_k/\sigma_k > \|s_k\|_2 = \delta_k, \tag{2.10}$$

from which it follows (by standard theory on the relationship between trust region subproblem solutions and their corresponding dual variables [23, Chap. 7]) that $\lambda_{k+1} \leq \lambda_k$. This, along with (2.9) and (2.10), implies that

$$\lambda_{k+1} \leq \lambda_k = \sigma_k\delta_{k+1} = \sigma_{k+1}\|s_{k+1}\|_2,$$

from which it follows that $(k+1) \notin \mathcal{E}$.

2. Suppose $\Delta_k < \lambda_k/\sigma_k$. It then follows from (2.9) and Step 16 that

$$\|s_{k+1}\|_2 = \delta_{k+1} \leftarrow \Delta_k = \Delta_{k+1}.$$

If $\rho_{k+1} \geq \eta$, then it follows that $(k+1) \in \mathcal{A}_\Delta \subseteq \mathcal{A}$. Otherwise, $\rho_{k+1} < \eta$, from which it follows that $(k+1) \in \mathcal{C}$. Hence, in either case $(k+1) \notin \mathcal{E}$.

Overall, we have shown that if $k \in \mathcal{C} \cup \mathcal{E}$, then $(k+1) \notin \mathcal{E}$, as desired. □

Next, we prove that if the dual variable for the trust region constraint is sufficiently large, then the constraint is active and the corresponding trial step yields a reduction in the objective function that is large relative to the trial step norm.

Lemma 7. *For any $k \in \mathbb{N}_+$, if the trial step s_k and dual variable λ_k satisfy*

$$\lambda_k \geq 2g_{Lip} + H_{max} + 2\eta\|s_k\|_2, \tag{2.11}$$

then $\|s_k\|_2 = \delta_k$ and $\rho_k \geq \eta$.

Proof. By the definition of the objective function model q_k, there exists a point $\bar{x}_k \in \mathbb{R}^n$ on the line segment $[x_k, x_k + s_k]$ such that

$$\begin{aligned} q_k(s_k) - f(x_k + s_k) &= (g_k - g(\bar{x}_k))^T s_k + \tfrac{1}{2} s_k^T H_k s_k \\ &\geq -\|g_k - g(\bar{x}_k)\|_2 \|s_k\|_2 - \tfrac{1}{2}\|H_k\|_2 \|s_k\|_2^2. \end{aligned} \tag{2.12}$$

Hence, from Lemma 2, (2.1b), (2.12), and the fact that (2.11) and (2.1c) imply that $\|s_k\|_2 = \delta_k > 0$, it follows that

$$\begin{aligned} f_k - f(x_k + s_k) &= f_k - q_k(s_k) + q_k(s_k) - f(x_k + s_k) \\ &\geq \tfrac{1}{2}\lambda_k \|s_k\|_2^2 - \|g_k - g(\bar{x}_k)\|_2 \|s_k\|_2 - \tfrac{1}{2}\|H_k\|_2 \|s_k\|_2^2 \\ &\geq (\tfrac{1}{2}\lambda_k - g_{Lip} - \tfrac{1}{2}H_{max})\|s_k\|_2^2 \\ &\geq \eta\|s_k\|_2^3, \end{aligned}$$

as desired. □

We may now combine previous results to show that if the algorithm were only to compute contraction steps from some iteration onward, then the sequences of trust region

radii and dual variables would converge to zero and infinity, respectively.

Lemma 8. *If $k \in \mathcal{C}$ for all sufficiently large $k \in \mathbb{N}_+$, then $\{\delta_k\} \to 0$ and $\{\lambda_k\} \to \infty$.*

Proof. Assume, without loss of generality, that $k \in \mathcal{C}$ for all $k \in \mathbb{N}_+$. It then follows from Lemma 3 that $\{\delta_k\}$ is monotonically strictly decreasing and $\{\lambda_k\}$ is monotonically nondecreasing. Combining this former fact with the fact that $\{\delta_k\}$ is bounded below by zero, we have that $\{\delta_k\}$ converges.

We may now observe that if Step 38 is reached infinitely often, then, clearly, $\{\delta_k\} \to 0$, from which it follows by standard trust region theory [23, Chap. 7] that $\{\lambda_k\} \to \infty$. Therefore, to complete the proof, let us assume that this update does not occur infinitely often, i.e., that there exists $k_{\mathcal{C}} \in \mathbb{N}_+$ such that Step 28, 31, or 36 is reached for all $k \geq k_{\mathcal{C}}$. In fact, we claim that we may assume without loss of generality that Step 28 or 36 is reached for all $k \geq k_{\mathcal{C}}$. We prove this claim in the following manner. Suppose that, for some $k \geq k_{\mathcal{C}}$, Step 31 is reached and the algorithm sets $\delta_{k+1} \leftarrow \|s\|_2$ where (λ, s) is computed by Step 30. Then, it follows that $\lambda_{k+1} \leftarrow \lambda$ and $s_{k+1} \leftarrow s$ where $\lambda_{k+1} \geq \underline{\sigma}\|s_{k+1}\|_2$. Therefore, during iteration $(k+1) \in \mathcal{C}$, it follows that the condition in Step 23 will test false, implying that the algorithm will set $\lambda_{k+2} > \lambda_{k+1}$ in Step 33. Since $\{\delta_k\}$ is monotonically strictly decreasing and $\{\lambda_k\}$ is monotonically nondecreasing, it follows that $\{\lambda_k/\|s_k\|_2\}$ is monotonically strictly increasing, implying that the condition in Step 23 will test false in all subsequent iterations, i.e., Step 31 will not be reached in any subsequent iteration. Overall, this analysis proves that we may assume without loss of generality that Step 28 or 36 is reached for all $k \geq k_{\mathcal{C}}$.

Consider iteration $k_{\mathcal{C}}$. If $\lambda_{k_{\mathcal{C}}} = 0$, then the condition in Step 23 tests true, in which case Step 25 will set $\lambda > 0$ and Step 28 will be reached so that $\lambda_{k_{\mathcal{C}}+1} = \lambda > 0$. On the other hand, $\lambda_{k_{\mathcal{C}}} > 0$ implies from Steps 25 and 33 that the algorithm will set $\lambda > 0$, which, since either Step 28 or 36 will be reached, implies that $\lambda_{k_{\mathcal{C}}+1} = \lambda > 0$. In either case, we have shown that $\lambda_{k_{\mathcal{C}}+1} > 0$, meaning that $\lambda_{k+1} \geq \min\{\lambda_k + (\underline{\sigma}\|g_k\|_2)^{1/2}, \gamma_\lambda \lambda_k\} > \lambda_k$ for all $k \geq k_{\mathcal{C}} + 1$. Moreover, since $k \in \mathcal{C}$ for all $k \geq k_{\mathcal{C}}$, we have $x_k = x_{k_{\mathcal{C}}}$ (and so $g_k = g_{k_{\mathcal{C}}}$) for all $k \geq k_{\mathcal{C}}$, which implies that, in fact, $\{\lambda_k\} \to \infty$. It now follows by standard trust region theory [23, Chap. 7] that $\|s_k\|_2 = \delta_k > 0$ for all $k > k_{\mathcal{C}}$, and, in addition, that $\{\delta_k\} \to 0$, as desired. □

We now prove that the set of accepted steps is infinite.

Lemma 9. *The set $\mathcal{A}$ has infinite cardinality.*

Proof. To derive a contradiction, suppose that $|\mathcal{A}| < \infty$. We claim that this implies $|\mathcal{C}| = \infty$. Indeed, if $|\mathcal{C}| < \infty$, then there exists some $k_{\mathcal{E}} \in \mathbb{N}_+$ such that $k \in \mathcal{E}$ for all $k \geq k_{\mathcal{E}}$, which contradicts Lemma 6. Thus, $|\mathcal{C}| = \infty$. Combining this with the result of Lemma 6, we conclude that there exists some $k_{\mathcal{C}} \in \mathbb{N}_+$ such that $k \in \mathcal{C}$ for all $k \geq k_{\mathcal{C}}$. From here, it follows from Steps 10 and 11 that $x_k = x_{k_{\mathcal{C}}}$ and $\rho_k < \eta$ for all $k \geq k_{\mathcal{C}}$, and from Lemma 8 that $\{\|s_k\|_2\} \leq \{\delta_k\} \to 0$ and $\{\lambda_k\} \to \infty$. In combination with Lemma 7, we conclude that there exists some $k \geq k_{\mathcal{C}}$ such that $\rho_k \geq \eta$, which contradicts the fact that $k \in \mathcal{C}$ for all $k \geq k_{\mathcal{C}}$. Having arrived at a contradiction under the supposition that $|\mathcal{A}| < \infty$, the result follows. □

We now prove that the elements of the sequence $\{\Delta_k\}$ are equal for all sufficiently large $k \in \mathbb{N}_+$, the set $\mathcal{A}_\Delta$ has finite cardinality, and there exists a uniform upper bound on the norms of the trial steps.

Lemma 10. *There exists a scalar constant $\Delta \in \mathbb{R}_{++}$ such that $\Delta_k = \Delta$ for all sufficiently large $k \in \mathbb{N}_+$, the set $\mathcal{A}_\Delta$ has finite cardinality, and there exists a scalar constant $s_{max} \in \mathbb{R}_{++}$ such that $\|s_k\|_2 \leq s_{max}$ for all $k \in \mathbb{N}_+$.*

Proof. For all $k \in \mathcal{A}$, we have $\rho_k \geq \eta$, which implies by Step 6 that

$$f(x_k) - f(x_{k+1}) \geq \eta \|s_k\|_2^3.$$

Combining this with Lemma 9 and the facts that the sequence $\{f_k\}$ is monotonically decreasing and f is bounded below, it follows that $\{s_k\}_{k \in \mathcal{A}} \to 0$; in particular, there exists $k_{\mathcal{A}} \in \mathbb{N}_+$ such that, for all $k \in \mathcal{A}$ with $k \geq k_{\mathcal{A}}$, we have

$$\gamma_{\mathrm{E}} \|s_k\|_2 \leq \Delta_0 \leq \Delta_k, \tag{2.13}$$

where the latter inequality follows from Lemma 4. It now follows from (2.13) and the updates in Steps 7, 12, and 16 that $\Delta_{k+1} \leftarrow \Delta_k$ for all $k \geq k_{\mathcal{A}}$, which proves the first part of the lemma. Next, we may observe that (2.13) also implies that $\|s_k\|_2 < \Delta_k$ for all $k \in \mathcal{A}$ with $k \geq k_{\mathcal{A}}$, from which it follows that $k \notin \mathcal{A}_\Delta$. This proves the second part of the lemma. Finally, the last part of the lemma follows from the first part and the fact that Lemma 4 ensures $\|s_k\|_2 \leq \delta_k \leq \Delta_k = \Delta$ for all sufficiently large $k \in \mathbb{N}_+$. □

Our next result ensures a lower bound on the trust region radii if the gradient sequence is asymptotically bounded away from zero.

Lemma 11. *If there exists a scalar constant $g_{min} > 0$ such that*

$$\|g_k\|_2 \geq g_{min} \quad \text{for all} \quad k \in \mathbb{N}_+, \tag{2.14}$$

then there exists a scalar constant $\delta_{min} > 0$ such that $\delta_k \geq \delta_{min}$ for all $k \in \mathbb{N}_+$.

Proof. If $|\mathcal{C}| < \infty$, then the result follows as a consequence of Lemma 5. Therefore, we may proceed under the assumption that $|\mathcal{C}| = \infty$.

We claim that there exists $\delta_{thresh} > 0$ such that, if $k \in \mathcal{C}$, then $\delta_k \geq \delta_{thresh}$. Indeed, as in the proof of Lemma 7 and by Lemma 2, we have for all $k \in \mathbb{N}_+$ that there exists some $\bar{x}_k$ on the line segment $[x_k, x_k + s_k]$ such that

$$\begin{aligned} & f_k - f(x_k + s_k) \\ = \; & f_k - q_k(s_k) + q_k(s_k) - f(x_k + s_k) \\ \geq \; & \tfrac{1}{2}\|g_k\|_2 \min\left\{\delta_k, \frac{\|g_k\|_2}{\|H_k\|_2}\right\} - \|g_k - g(\bar{x}_k)\|_2\|s_k\|_2 - \tfrac{1}{2}\|H_k\|_2\|s_k\|_2^2. \end{aligned} \tag{2.15}$$

Consequently, we have $\rho_k \geq \eta$ if $\delta_k \in (0, g_{min}/H_{max}]$ is sufficiently small such that

$$\tfrac{1}{2} g_{min}\delta_k - (g_{Lip} + \tfrac{1}{2}H_{max})\delta_k^2 \geq \eta\delta_k^3 \geq \eta\|s_k\|_2^3.$$

This fact implies the existence of a positive threshold $\delta_{thresh} \in (0, g_{min}/H_{max}]$ such that, for any $k \in \mathbb{N}_+$ with $\delta_k \in (0, \delta_{thresh})$, we have $\rho_k \geq \eta$. Hence, as desired, we have proved that if $k \in \mathcal{C}$, then $\delta_k \geq \delta_{thresh}$.

Now suppose that $k \in \mathcal{C}$ and consider the update for δ_{k+1} in Step 13, which calls the CONTRACT subroutine. If Step 28 is reached, then it follows that

$$\delta_{k+1} \leftarrow \|s\|_2 \geq \frac{\lambda}{\overline{\sigma}} = \frac{\lambda_k + (\underline{\sigma}\|g_k\|_2)^{1/2}}{\overline{\sigma}} \geq \frac{(\underline{\sigma} g_{min})^{1/2}}{\overline{\sigma}},$$

while if Step 36 or 38 is reached, then it follows that

$$\delta_{k+1} \geq \gamma_{\mathrm{C}}\|s_k\|_2 = \gamma_{\mathrm{C}}\delta_k \geq \gamma_{\mathrm{C}}\delta_{thresh},$$

where we have used the fact that $\lambda_k > 0$ (see Step 32) implies by (2.1c) that $\|s_k\|_2 = \delta_k$. In all of these cases, we have proved a constant lower bound for δ_{k+1}. All that remains is to consider the value for δ_{k+1} obtained if Step 31 is reached. For this case, let us use (λ, s) to denote the values obtained in Step 30 while we use $\hat{s}$ to denote the solution that was obtained in Step 26. Since $\lambda \in (\lambda_k, \hat{\lambda})$, we have that $\|s\|_2 \geq \|\hat{s}\|_2$. Now let $H_k = V_k \Xi_k V_k^T$ where V_k is an orthonormal matrix of eigenvectors and $\Xi_k = \operatorname{diag}(\xi_{k,1}, \ldots, \xi_{k,n})$ with $\xi_{k,1} \leq \cdots \leq \xi_{k,n}$ is a diagonal matrix of eigenvalues of H_k. Since $\hat{\lambda} > \lambda_k$, the matrix $H_k + \hat{\lambda} I$ is invertible and

$$\|\hat{s}\|_2^2 = \|V_k(\Xi_k + \hat{\lambda} I)^{-1} V_k^T g_k\|_2^2 = g_k^T V_k (\Xi_k + \hat{\lambda} I)^{-2} V_k^T g_k,$$

which implies from the orthonormality of V_k, Step 23, and Lemma 10 that

$$\begin{aligned}
\frac{\|\hat{s}\|_2^2}{\|g_k\|_2^2} &= \frac{g_k^T V_k (\Xi_k + \hat{\lambda} I)^{-2} V_k^T g_k}{\|V_k^T g_k\|_2^2} \\
&\geq \left(\xi_{k,n} + \lambda_k + (\underline{\sigma}\|g_k\|_2)^{1/2}\right)^{-2} \\
&\geq \left(\xi_{k,n} + \underline{\sigma}\|s_k\|_2 + (\underline{\sigma}\|g_k\|_2)^{1/2}\right)^{-2} \\
&\geq \left(H_{max} + \underline{\sigma} s_{max} + (\underline{\sigma} g_{max})^{1/2}\right)^{-2}.
\end{aligned}$$

Hence, under the conditions of the lemma, there exists a constant $s_{min} > 0$ such that, for all such $k \in \mathcal{C}$, we have $\delta_{k+1} \leftarrow \|s\|_2 \geq \|\hat{s}\|_2 \geq s_{min}$. Combining all of the cases in the above analysis, we have shown that, for all $k \in \mathcal{C}$, we have

$$\delta_{k+1} \geq \min\left\{ \frac{(\underline{\sigma} g_{min})^{1/2}}{\overline{\sigma}}, \gamma_C \delta_{thresh}, s_{min} \right\} > 0.$$

Overall, the result follows by combining the constant positive lower bound for δ_{k+1} for $k \in \mathcal{C}$ provided by the previous paragraph with the fact that Lemma 5 ensures that $\delta_{k+1} \geq \delta_k$ for all $k \in \mathcal{A} \cup \mathcal{E}$. □

We now prove a standard result for trust region algorithms showing that the limit inferior of the norms of the gradients of the objective is equal to zero.

Lemma 12. *There holds*

$$\liminf_{k \in \mathbb{N}_+, k \to \infty} \|g_k\|_2 = 0.$$

Proof. Suppose that there exists a scalar constant $g_{min} > 0$ such that (2.14) holds. Then, by Lemma 11, there exists a scalar constant $\delta_{min} > 0$ where $\delta_k \geq \delta_{min} > 0$ for all $k \in \mathbb{N}_+$. This fact, along with (2.6), (2.14), and boundedness of $\{\|H_k\|_2\}$, implies that there exists a scalar constant $s_{min} > 0$ such that $\|s_k\|_2 \geq s_{min}$ for all $k \in \mathbb{N}_+$. On the other hand, for all $k \in \mathcal{A}$ we have $\rho_k \geq \eta$, which implies that $f_k - f_{k+1} \geq \eta \|s_k\|_2^3$. Since f is bounded below on $\mathbb{R}^n$ and Lemma 9 ensures that $|\mathcal{A}| = \infty$, this implies that $\{s_k\}_{k \in \mathcal{A}} \to 0$, contradicting the existence of $s_{min} > 0$. Overall, there cannot exist $g_{min} > 0$ such that (2.14) holds, so the result follows. □

We close with our main global convergence result of this subsection.

Theorem 3. *Under Assumptions 3 and 4, it holds that*

$$\lim_{k \in \mathbb{N}_+, k \to \infty} \|g_k\|_2 = 0. \tag{2.16}$$

Proof. To reach a contradiction, suppose that (2.16) does not hold. Combining this with the results of Lemmas 9 and 12, it follows that there exists an infinite subsequence $\{t_i\} \subseteq \mathcal{A}$ (indexed over $i \in \mathbb{N}_+$) such that, for some $\epsilon > 0$ and all $i \in \mathbb{N}_+$, we have $\|g_{t_i}\|_2 \geq 2\epsilon > 0$. Additionally, Lemmas 9 and 12 imply that there exists an infinite subsequence $\{\ell_i\} \subseteq \mathcal{A}$ (indexed over $i \in \mathbb{N}_+$) such that, for all $i \in \mathbb{N}_+$ and $k \in \mathbb{N}_+$ with $t_i \leq k < \ell_i$, we have

$$\|g_k\|_2 \geq \epsilon \quad \text{and} \quad \|g_{\ell_i}\|_2 < \epsilon. \tag{2.17}$$

We now restrict our attention to indices in the infinite index set

$$\mathcal{K} := \{k \in \mathcal{A} : t_i \leq k < \ell_i \text{ for some } i \in \mathbb{N}_+\}.$$

Observe from (2.17) that, for all $k \in \mathcal{K}$, we have $\|g_k\|_2 \geq \epsilon$. Hence, by Lemma 1, we have for all $k \in \mathcal{K} \subseteq \mathcal{A}$ that

$$f_k - f_{k+1} \geq \eta \|s_k\|_2^3 \geq \eta \left(\min \left\{ \delta_k, \frac{\epsilon}{H_{max}} \right\} \right)^3. \tag{2.18}$$

Since $\{f_k\}$ is monotonically decreasing and bounded below, we know that $f_k \to \underline{f}$ for some

$\underline{f} \in \mathbb{R}$, which when combined with (2.18) shows that

$$\lim_{k \in \mathcal{K}, k \to \infty} \delta_k = 0. \tag{2.19}$$

Using this fact and (2.15), we have for all sufficiently large $k \in \mathcal{K}$ that

$$\begin{aligned} f_k - f_{k+1} &\geq \tfrac{1}{2}\|g_k\|_2 \min\left\{\delta_k, \frac{\|g_k\|_2}{\|H_k\|_2}\right\} - (g_{Lip} + \tfrac{1}{2}H_{max})\|s_k\|_2^2 \\ &\geq \tfrac{1}{2}\epsilon \min\left\{\delta_k, \frac{\epsilon}{H_{max}}\right\} - (g_{Lip} + \tfrac{1}{2}H_{max})\|s_k\|_2^2 \\ &\geq \tfrac{1}{2}\epsilon\delta_k - (g_{Lip} + \tfrac{1}{2}H_{max})\delta_k^2. \end{aligned}$$

From this inequality and (2.19), it follows that $f_k - f_{k+1} \geq \epsilon\delta_k/4$ for all sufficiently large $k \in \mathcal{K}$. Consequently, we have for all sufficiently large $i \in \mathbb{N}_+$ that

$$\begin{aligned} \|x_{t_i} - x_{\ell_i}\|_2 &\leq \sum_{k \in \mathcal{K}, k = t_i}^{\ell_i - 1} \|x_k - x_{k+1}\|_2 \\ &\leq \sum_{k \in \mathcal{K}, k = t_i}^{\ell_i - 1} \delta_k \leq \sum_{k \in \mathcal{K}, k = t_i}^{\ell_i - 1} \tfrac{4}{\epsilon}(f_k - f_{k+1}) = \tfrac{4}{\epsilon}(f_{t_i} - f_{\ell_i}). \end{aligned}$$

Since $\{f_{t_i} - f_{\ell_i}\} \to 0$, this implies that $\{\|x_{t_i} - x_{\ell_i}\|_2\} \to 0$, which, in turn, implies that $\{\|g_{t_i} - g_{\ell_i}\|_2\} \to 0$. However, this is a contradiction since, for any $i \in \mathbb{N}_+$, we have $\|g_{t_i} - g_{\ell_i}\|_2 \geq \epsilon$ by the definitions of $\{t_i\}$ and $\{\ell_i\}$. Overall, we conclude that our initial supposition must be false, implying that (2.16) holds. □

2.2.2 Worst-case iteration complexity to approximate first-order stationarity

Our goal in this subsection is to prove a worst-case upper bound on the number of iterations required for TRACE to reduce the norm of the gradient of the objective below a prescribed positive threshold, namely $\epsilon \in (0, \infty)$. For this purpose, in addition to Assumptions 3 and 4, we add the following assumption about the objective function and the sequences of iterates and computed trial steps.

Assumption 5. *The Hessian function H is Lipschitz continuous on a path defined by the sequence of iterates and trial steps; in particular, it is Lipschitz continuous with a scalar*

Lipschitz constant $H_{Lip} > 0$ *on the set* $\{x_k + \tau s_k : k \in \mathbb{N}_+, \tau \in [0,1]\}$.

We begin our analysis in this subsection by providing a refinement of Lemma 7 under the additional assumption made in this subsection.

Lemma 13. *For any* $k \in \mathbb{N}_+$, *if the trial step* s_k *and dual variable* λ_k *satisfy*

$$\lambda_k \geq (H_{Lip} + 2\eta)\|s_k\|_2, \tag{2.20}$$

then $\|s_k\|_2 = \delta_k$ *and* $\rho_k \geq \eta$.

Proof. For all $k \in \mathbb{N}_+$, there exists $\bar{x}_k$ on the line segment $[x_k, x_k + s_k]$ such that

$$q_k(s_k) - f(x_k + s_k) = \tfrac{1}{2} s_k^T (H_k - H(\bar{x}_k)) s_k \geq -\tfrac{1}{2} H_{Lip} \|s_k\|_2^3. \tag{2.21}$$

Hence, Lemma 2, (2.21), and (2.1b) imply that, for any $k \in \mathbb{N}_+$,

$$\begin{aligned} f_k - f(x_k + s_k) &= f_k - q_k(s_k) + q_k(s_k) - f(x_k + s_k) \\ &\geq \tfrac{1}{2}\lambda_k \|s_k\|_2^2 - \tfrac{1}{2} H_{Lip} \|s_k\|_2^3. \end{aligned}$$

This and (2.20) imply $\rho_k \geq \eta$, whereas (2.20) and (2.1c) imply $\|s_k\|_2 = \delta_k$. □

We now prove bounds for a critical ratio that hold after any contraction.

Lemma 14. *For any* $k \in \mathbb{N}_+$, *if* $k \in \mathcal{C}$, *then*

$$\underline{\sigma} \leq \frac{\lambda_{k+1}}{\|s_{k+1}\|_2} \leq \max\left\{\overline{\sigma}, \left(\frac{\gamma_\lambda}{\gamma_C}\right)\frac{\lambda_k}{\|s_k\|_2}\right\}.$$

Proof. Let $k \in \mathcal{C}$ and consider the four cases that may occur within CONTRACT. The first two correspond to situations in which the condition in Step 23 tests true.

1. Suppose that Step 28 is reached so that $\delta_{k+1} \leftarrow \|s\|_2$ where (λ, s) is computed in Steps 25–26. It follows that Step 19 will then produce the primal-dual pair (s_{k+1}, λ_{k+1}) solving $\mathcal{Q}_{k+1}$ such that $s_{k+1} = s$ and $\lambda_{k+1} = \lambda > 0$. Since the condition in Step 27 tested true, this implies that

$$\frac{\lambda_{k+1}}{\|s_{k+1}\|_2} = \frac{\lambda}{\|s\|_2} \leq \overline{\sigma}.$$

On the other hand, a lower bound for this ratio can be found using a similar technique as in the proof of Lemma 11; in particular, as in that proof (where (λ, s) from Steps 25–26 temporarily appeared as $(\hat{s}, \hat{\lambda})$), we have

$$\frac{\|s\|_2^2}{\|g_k\|_2^2} = \frac{g_k^T V_k (\Xi_k + \lambda I)^{-2} V_k^T g_k}{\|V_k^T g_k\|_2^2} \leq \left(\xi_{k,1} + \lambda_k + (\underline{\sigma}\|g_k\|_2)^{1/2}\right)^{-2}.$$

Hence, since $\lambda_k \geq \max\{0, -\xi_{k,1}\}$, we have that

$$\frac{\lambda_{k+1}}{\|s_{k+1}\|_2} = \frac{\lambda}{\|s\|_2} \geq \frac{(\lambda_k + (\underline{\sigma}\|g_k\|_2)^{1/2})(\xi_{k,1} + \lambda_k + (\underline{\sigma}\|g_k\|_2)^{1/2})}{\|g_k\|_2} \geq \underline{\sigma}.$$

2. Suppose that Step 31 is reached so that $\delta_{k+1} \leftarrow \|s\|_2$ where (λ, s) is computed in Step 30. Similarly to the previous case, it follows that Step 19 will produce the primal-dual pair (s_{k+1}, λ_{k+1}) solving $\mathcal{Q}_{k+1}$ such that $s_{k+1} = s$ and $\lambda_{k+1} = \lambda > 0$. Since Step 30 was reached, this implies that

$$\frac{\lambda_{k+1}}{\|s_{k+1}\|_2} = \frac{\lambda}{\|s\|_2} \quad \text{where} \quad \underline{\sigma} \leq \frac{\lambda}{\|s\|_2} \leq \overline{\sigma}.$$

The other two cases that may occur within CONTRACT correspond to situations in which the condition in Step 23 tests false, in which case $\lambda_k > 0$ and the pair (λ, s) is computed in Step 33–34. This means, in particular, that

$$\underline{\sigma} \leq \frac{\lambda_k}{\|s_k\|_2} \leq \frac{\lambda}{\|s\|_2}, \tag{2.22}$$

where the latter inequality follows since $\lambda = \gamma_\lambda \lambda_k > \lambda_k$, which, in turn, implies by standard trust region theory [23, Chap. 7] that $\|s\|_2 < \|s_k\|_2$. We now consider the two cases that may occur under this scenario.

3. Suppose that Step 36 is reached so that $\delta_{k+1} \leftarrow \|s\|_2$. It follows that Step 19 will produce the primal-dual pair (s_{k+1}, λ_{k+1}) solving $\mathcal{Q}_{k+1}$ such that $s_{k+1} = s$ and $\lambda_{k+1} = \lambda$. In conjunction with (2.22), we may then observe that

$$\underline{\sigma} \leq \frac{\lambda_{k+1}}{\|s_{k+1}\|_2} = \frac{\lambda}{\|s\|_2} = \frac{\gamma_\lambda \lambda_k}{\|s\|_2} \leq \frac{\gamma_\lambda \lambda_k}{\gamma_C \|s_k\|_2}.$$

4. Suppose that Step 38 is reached so that $\delta_{k+1} \leftarrow \gamma_C \|s_k\|_2$. It follows that Step 19 will

produce the primal-dual pair (s_{k+1}, λ_{k+1}) solving $\mathcal{Q}_{k+1}$ such that $\lambda_{k+1} > 0$ since $\|s_{k+1}\|_2 = \delta_{k+1} = \gamma_C \|s_k\|_2$. Also, since the condition in Step 35 tested false, we conclude by Lemma 3 that $\lambda_k \leq \lambda_{k+1} \leq \lambda = \gamma_\lambda \lambda_k$. Hence, with (2.22),

$$\underline{\sigma} < \frac{\underline{\sigma}}{\gamma_C} \leq \frac{\lambda_k}{\gamma_C \|s_k\|_2} \leq \frac{\lambda_{k+1}}{\|s_{k+1}\|_2} \leq \frac{\gamma_\lambda \lambda_k}{\gamma_C \|s_k\|_2}.$$

The result follows since we have obtained the desired inequalities in all cases. □

The results of the two preceding lemmas can be combined to prove that the sequence $\{\sigma_k\}$ is bounded above.

Lemma 15. *There exists a scalar constant $\sigma_{max} > 0$ such that, for all $k \in \mathbb{N}_+$,*

$$\sigma_k \leq \sigma_{max}.$$

Proof. Observe by Steps 9, 18, and 21 that $\{\sigma_k\}$ is monotonically nondecreasing. Moreover, Lemma 10 ensures the existence of $k_{\mathcal{A}} \in \mathbb{N}_+$ such that, if $k \in \mathcal{A}$ with $k \geq k_{\mathcal{A}}$, then $k \in \mathcal{A}_\sigma$. We now consider three cases for any $k \in \mathbb{N}_+$ with $k \geq k_{\mathcal{A}}$.

1. If $k \in \mathcal{A}_\sigma \subseteq \mathcal{A}$, then $\lambda_k \leq \sigma_k \|s_k\|_2$, which implies by Step 9 that $\sigma_{k+1} \leftarrow \sigma_k$.

2. If $k \in \mathcal{C}$, then $\rho_k < \eta$, which implies by Lemma 13 that

$$\lambda_k < (H_{Lip} + 2\eta)\|s_k\|_2.$$

   Hence, by Step 21 and Lemma 14, it follows that

$$\sigma_{k+1} \leftarrow \max\left\{\sigma_k, \frac{\lambda_{k+1}}{\|s_{k+1}\|_2}\right\} \leq \max\left\{\sigma_k, \overline{\sigma}, \left(\frac{\gamma_\lambda}{\gamma_C}\right)(H_{Lip} + 2\eta)\right\}.$$

3. If $k \in \mathcal{E}$, then Step 18 implies that $\sigma_{k+1} \leftarrow \sigma_k$.

Combining the results of these three cases, the desired result follows. □

We now establish a lower bound on the norm of any accepted step in $\mathcal{A}_\sigma$.

Lemma 16. *For all $k \in \mathcal{A}_\sigma$, the accepted step s_k satisfies*

$$\|s_k\|_2 \geq \left(\tfrac{1}{2}H_{Lip} + \sigma_{max}\right)^{-1/2} \|g_{k+1}\|_2^{1/2}.$$

Proof. For all $k \in \mathcal{A}$, it follows from (2.1a) that

$$\begin{aligned}\|g_{k+1}\|_2 &= \|g(x_k + s_k) - (g_k + (H_k + \lambda_k I)s_k)\|_2 \\ &\leq \|g(x_k + s_k) - (g_k + H_k s_k)\|_2 + \lambda_k \|s_k\|_2.\end{aligned}$$

By Taylor's theorem, the first term on the right-hand side of (2.2.2) satisfies

$$\begin{aligned}\|g(x_k + s_k) - (g_k + H_k s_k)\|_2 &\leq \left\| \int_0^1 (H(x_k + \tau s_k) - H_k) s_k d\tau \right\|_2 \\ &\leq \int_0^1 \|H(x_k + \tau s_k) - H_k\|_2 \, d\tau \|s_k\|_2 \\ &\leq \int_0^1 \tau d\tau H_{Lip} \|s_k\|_2^2 \\ &= \tfrac{1}{2} H_{Lip} \|s_k\|_2^2,\end{aligned}$$

which, with (2.2.2) and the fact that $\lambda_k \leq \sigma_k \|s_k\|_2$ for all $k \in \mathcal{A}_\sigma$, implies that

$$\begin{aligned}\|g_{k+1}\|_2 &\leq \tfrac{1}{2} H_{Lip} \|s_k\|_2^2 + \left(\frac{\lambda_k}{\|s_k\|_2} \right) \|s_k\|_2^2 \\ &\leq \left(\tfrac{1}{2} H_{Lip} + \sigma_k \right) \|s_k\|_2^2.\end{aligned}$$

This, along with Lemma 15, implies the result. □

We are now prepared to prove a worst-case upper bound on the total number of certain accepted steps that may occur in iterations in which the norm of the gradient of the objective is above a prescribed positive threshold.

Lemma 17. *For a scalar $\epsilon \in (0, \infty)$, the total number of elements in the index set*

$$\mathcal{K}_\epsilon := \{k \in \mathbb{N}_+ : k \geq 1, \ (k-1) \in \mathcal{A}_\sigma, \ \|g_k\|_2 > \epsilon\}$$

is at most

$$\left\lfloor \left(\frac{f_0 - f_{min}}{\eta \left(\frac{1}{2} H_{Lip} + \sigma_{max} \right)^{-3/2}} \right) \epsilon^{-3/2} \right\rfloor =: K_\sigma(\epsilon) \geq 0. \tag{2.23}$$

Proof. By Lemma 16, we have for all $k \in \mathcal{K}_\epsilon$ that

$$\begin{aligned} f_{k-1} - f_k &\geq \eta \|s_{k-1}\|_2^3 \\ &\geq \eta \left(\tfrac{1}{2} H_{Lip} + \sigma_{max}\right)^{-3/2} \|g_k\|_2^{3/2} \\ &\geq \eta \left(\tfrac{1}{2} H_{Lip} + \sigma_{max}\right)^{-3/2} \epsilon^{3/2}. \end{aligned}$$

In addition, we have by Theorem 3 that $|\mathcal{K}_\epsilon| < \infty$. Hence, the reduction in f obtained up to the largest index in $\mathcal{K}_\epsilon$, call it $\bar{k}_\epsilon$, satisfies

$$\begin{aligned} f_0 - f_{\bar{k}_\epsilon} &= \sum_{k=1}^{\bar{k}_\epsilon} (f_{k-1} - f_k) \\ &\geq \sum_{k \in \mathcal{K}_\epsilon} (f_{k-1} - f_k) \\ &\geq |\mathcal{K}_\epsilon| \eta \left(\tfrac{1}{2} H_{Lip} + \sigma_{max}\right)^{-3/2} \epsilon^{3/2}. \end{aligned}$$

Rearranging this inequality to yield an upper bound for $|\mathcal{K}_\epsilon|$ and using the fact that $f_0 - f_{min} \geq f_0 - f_{\bar{k}_\epsilon}$, we obtain the desired result. □

In order to prove a result similar to Lemma 17 for the *total* number of iterations with $\|g_k\|_2 > \epsilon$, we require an upper bound on the cardinality of the set $\mathcal{A}_\Delta$, as well as an upper bound on the total number of contraction and expansion iterations that may occur until the next accepted step. We obtain the first such bound with the following refinement for the second part of the result of Lemma 10.

Lemma 18. *The cardinality of the set $\mathcal{A}_\Delta$ is bounded above by*

$$\left\lfloor \frac{f_0 - f_{min}}{\eta \Delta_0^3} \right\rfloor =: K_\Delta \geq 0. \tag{2.24}$$

Proof. For all $k \in \mathcal{A}_\Delta$, it follows along with Lemma 4 that

$$f_k - f_{k+1} \geq \eta \|s_k\|_2^3 = \eta \Delta_k^3 \geq \eta \Delta_0^3.$$

Hence, we have that

$$f_0 - f_{min} \geq \sum_{k=0}^{\infty} (f_k - f_{k+1}) \geq \sum_{k \in \mathcal{A}_\Delta} (f_k - f_{k+1}) \geq |\mathcal{A}_\Delta| \eta \Delta_0^3,$$

from which the desired result follows. □

Now, for the purpose of deriving an upper bound on the numbers of contraction and expansion iterations that may occur until the next accepted step, let us define, for a given $\hat{k} \in \mathcal{A} \cup \{0\}$, the iteration number and corresponding set

$$k_{\mathcal{A}}(\hat{k}) := \min\{k \in \mathcal{A} : k > \hat{k}\}$$
$$\text{and} \quad \mathcal{I}(\hat{k}) := \{k \in \mathbb{N}_+ : \hat{k} < k < k_{\mathcal{A}}(\hat{k})\},$$

i.e, we let $k_{\mathcal{A}}(\hat{k})$ be the smallest of all iteration numbers in $\mathcal{A}$ that is strictly larger than $\hat{k}$, and we let $\mathcal{I}(\hat{k})$ be the set of intermediate iteration numbers between $\hat{k}$ and $k_{\mathcal{A}}(\hat{k})$. Using this notation, the following result shows that the number of expansion iterations between the first iteration and the first accepted step, or between consecutive accepted steps, is never greater than one. Moreover, when such an expansion iteration occurs, it must take place immediately.

Lemma 19. *For any $\hat{k} \in \mathbb{N}_+$, if $\hat{k} \in \mathcal{A} \cup \{0\}$, then $\mathcal{E} \cap \mathcal{I}(\hat{k}) \subseteq \{\hat{k}+1\}$.*

Proof. By the definition of $k_{\mathcal{A}}(\hat{k})$, we have under the conditions of the lemma that $\mathcal{I}(\hat{k}) \cap \mathcal{A} = \emptyset$, which means that $\mathcal{I}(\hat{k}) \subseteq \mathcal{C} \cup \mathcal{E}$. It then follows from Lemma 6 that $(k+1) \notin \mathcal{E}$ for all $k \in \mathcal{I}(\hat{k})$, so that $\mathcal{E} \cap \mathcal{I}(\hat{k}) \subseteq \{\hat{k}+1\}$, as desired. □

We now turn our attention to contraction steps that may occur between consecutive accepted steps. We first show that, when a critical ratio is bounded below by $\underline{\sigma}$, then it must increase by a constant factor during a contraction.

Lemma 20. *For any $k \in \mathbb{N}_+$, if $k \in \mathcal{C}$ and $\lambda_k \geq \underline{\sigma}\|s_k\|_2$, then*

$$\frac{\lambda_{k+1}}{\|s_{k+1}\|_2} \geq \min\left\{\gamma_\lambda, \frac{1}{\gamma_{\text{C}}}\right\}\left(\frac{\lambda_k}{\|s_k\|_2}\right).$$

Proof. Since $k \in \mathcal{C}$ and $\lambda_k \geq \underline{\sigma}\|s_k\|_2 > 0$, it follows that the condition in Step 23 tests false. Hence, (λ, s) is computed in Steps 33–34 where $\lambda = \gamma_\lambda \lambda_k > \lambda_k$ and s solves $\mathcal{Q}_k(\lambda)$. We now consider the two cases that may occur in CONTRACT.

1. Suppose that Step 36 is reached, meaning that $\|s\|_2 \geq \gamma_{\text{C}}\|s_k\|_2$. It follows that Step 19 will produce the primal-dual pair (s_{k+1}, λ_{k+1}) solving $\mathcal{Q}_{k+1}$ such that (recall

Lemma 3) $\|s_{k+1}\|_2 = \delta_{k+1} < \delta_k = \|s_k\|_2$ and $\lambda_{k+1} = \gamma_\lambda \lambda_k$, i.e.,

$$\frac{\lambda_{k+1}}{\|s_{k+1}\|_2} > \frac{\gamma_\lambda \lambda_k}{\|s_k\|_2}. \tag{2.25}$$

2. Suppose that Step 38 is reached, meaning that $\|s\|_2 < \gamma_C \|s_k\|_2$. It follows that Step 19 will produce the primal-dual pair (s_{k+1}, λ_{k+1}) solving $\mathcal{Q}_{k+1}$ such that $\|s_{k+1}\|_2 = \delta_{k+1} = \gamma_C \|s_k\|_2$ and (recall Lemma 3) $\lambda_{k+1} \geq \lambda_k$. Consequently,

$$\frac{\lambda_{k+1}}{\|s_{k+1}\|_2} \geq \frac{\lambda_k}{\gamma_C \|s_k\|_2}. \tag{2.26}$$

The result now follows from (2.25) and (2.26). □

Our next result shows that the number of contraction steps between the first iteration and the first accepted step, or between consecutive accepted steps, is bounded above by a scalar constant.

Lemma 21. *For any $\hat{k} \in \mathbb{N}_+$, if $\hat{k} \in \mathcal{A} \cup \{0\}$, then*

$$|\mathcal{C} \cap \mathcal{I}(\hat{k})| \leq 1 + \left\lfloor \frac{1}{\log(\min\{\gamma_\lambda, \gamma_C^{-1}\})} \log\left(\frac{\sigma_{max}}{\underline{\sigma}}\right) \right\rfloor =: K_{\mathcal{C}} \geq 0.$$

Proof. The result holds trivially if $|\mathcal{C} \cap \mathcal{I}(\hat{k})| = 0$. Thus, we may assume $|\mathcal{C} \cap \mathcal{I}(\hat{k})| \geq 1$ and, by Lemma 19, may define the iteration number $k_{\mathcal{C}}(\hat{k}) \in \{\hat{k}+1, \hat{k}+2\}$ as the smallest element of $\mathcal{C} \cap \mathcal{I}(\hat{k})$. It follows along with Lemmas 3 and 4 that, for all $k \in \mathbb{N}_+$ with $k_{\mathcal{C}}(\hat{k}) + 1 \leq k \leq k_{\mathcal{A}}(\hat{k})$, we have

$$\|s_k\|_2 \leq \delta_k \leq \delta_{k_{\mathcal{C}}(\hat{k})+1} < \delta_{k_{\mathcal{C}}(\hat{k})} \leq \Delta_{k_{\mathcal{C}}(\hat{k})} \leq \Delta_{k_{\mathcal{A}}(\hat{k})}.$$

In particular, for $k = k_{\mathcal{A}}(\hat{k})$, this shows that $k_{\mathcal{A}}(\hat{k}) \in \mathcal{A}_\sigma$. Now, from Lemma 14,

$$\frac{\lambda_{k_{\mathcal{C}}(\hat{k})+1}}{\|s_{k_{\mathcal{C}}(\hat{k})+1}\|_2} \geq \underline{\sigma},$$

which, by the fact that $k_{\mathcal{A}}(\hat{k}) \in \mathcal{A}_\sigma$ and Lemmas 15 and 20, implies that

$$\sigma_{max} \geq \frac{\lambda_{k_{\mathcal{A}}(\hat{k})}}{\|s_{k_{\mathcal{A}}(\hat{k})}\|_2} \geq \underline{\sigma} \left(\min \left\{ \gamma_\lambda, \frac{1}{\gamma_{\mathrm{C}}} \right\} \right)^{k_{\mathcal{A}}(\hat{k}) - k_{\mathcal{C}}(\hat{k}) - 1},$$

from which it follows that

$$k_{\mathcal{A}}(\hat{k}) - k_{\mathcal{C}}(\hat{k}) \leq 1 + \frac{1}{\log(\min\{\gamma_\lambda, \gamma_{\mathrm{C}}^{-1}\})} \log \left(\frac{\sigma_{max}}{\underline{\sigma}} \right).$$

The desired result follows from this inequality since $|\mathcal{C} \cap \mathcal{I}(\hat{k})| = k_{\mathcal{A}}(\hat{k}) - k_{\mathcal{C}}(\hat{k})$. □

We are now prepared to prove our main complexity result of this subsection.

Theorem 4. *Under Assumptions 3, 4, and 5, for a scalar $\epsilon \in (0, \infty)$, the total number of elements in the index set*

$$\{k \in \mathbb{N}_+ : \|g_k\|_2 > \epsilon\}$$

is at most

$$K(\epsilon) := 1 + (K_\sigma(\epsilon) + K_\Delta)(1 + K_{\mathcal{C}}), \tag{2.27}$$

where $K_\sigma(\epsilon)$, K_Δ, and $K_{\mathcal{C}}$ are defined in Lemmas 17, 18, and 21, respectively. Consequently, we have that $K(\epsilon) = \mathcal{O}(\epsilon^{-3/2})$ as $\epsilon \searrow 0$.

Proof. Without loss of generality, we may assume that at least one iteration is performed. Lemmas 17 and 18 guarantee that the total number of elements in the index set $\{k \in \mathcal{A} : k \geq 1, \ \|g_k\|_2 > \epsilon\}$ is at most $K_\sigma(\epsilon) + K_\Delta$, where, immediately prior to each of the corresponding accepted steps, Lemmas 19 and 21 guarantee that at most $1 + K_{\mathcal{C}}$ expansion and contraction steps are performed. Also accounting for the first iteration, the desired result follows. □

2.2.3 Convergence and complexity to (approximate) second-order stationarity

We have established by Theorem 3 that any limit point of $\{x_k\}$ is a first-order stationary point for f, i.e., the gradient of f must be zero at any such point. Moreover, we have established by Theorem 4 that, in the worst case, the number of iterations required to compute an ϵ-stationary point (i.e., an iterate at which the norm of the gradient of f

is less than a prescribed tolerance $\epsilon \in (0, \infty)$) is $\mathcal{O}(\epsilon^{-3/2})$. Building on these results, our goal in this subsection is to analyze the global convergence properties and worst-case iteration complexity of TRACE toward (approximate) second-order stationary points. For our purposes in this section, we continue to make Assumptions 3, 4, and 5. In addition, we continue to use the notation introduced in Lemma 11, where, for all $k \in \mathbb{N}_+$, we let $H_k = V_k \Xi_k V_k^T$ where V_k is an orthonormal matrix of eigenvectors and $\Xi_k = \text{diag}(\xi_{k,1}, \ldots, \xi_{k,n})$ with $\xi_{k,1} \leq \cdots \leq \xi_{k,n}$ is a diagonal matrix of eigenvalues of H_k.

Along with Theorem 3, the following theorem establishes that the iterates of TRACE are driven toward second-order stationarity.

Theorem 5. *Under Assumptions 3, 4, and 5, it holds that*

$$\liminf_{k \in \mathbb{N}_+, k \to \infty} \xi_{k,1} \geq 0.$$

Proof. For all $k \in \mathcal{A}_\sigma$, we have $f_k - f_{k+1} \geq \eta \|s_k\|_2^3$. Hence, since $\{f_k\}$ is monotonically decreasing and bounded below, Lemma 9 yields $\{\|s_k\|_2\}_{k \in \mathcal{A}_\sigma} \to 0$. This, along with the fact that by Lemma 15 we have $\lambda_k \leq \sigma_k \|s_k\|_2 \leq \sigma_{max} \|s_k\|_2$ for all $k \in \mathcal{A}_\sigma$, implies that $\{\lambda_k\}_{k \in \mathcal{A}_\sigma} \to 0$. We may now recall from (2.1b) that $\xi_{k,1} + \lambda_k \geq 0$ for all $k \in \mathbb{N}_+$, which, along with the fact that $\{\lambda_k\}_{k \in \mathcal{A}_\sigma} \to 0$, yields

$$\liminf_{k \in \mathcal{A}_\sigma, k \to \infty} \xi_{k,1} \geq 0.$$

The result now follows from the above lower bound, the fact that Lemma 10 provides that $\mathcal{A}_\Delta$ has finite cardinality, and since $H_{k+1} = H_k$ for all $k \notin \mathcal{A}$. □

We now provide an upper bound on the number of accepted steps that occur when the leftmost eigenvalue of the Hessian matrix is below a negative threshold.

Lemma 22. *For a scalar $\epsilon \in (0, \infty)$, the total number of elements in the index set*

$$\mathcal{K}_{\epsilon,2} := \{k \in \mathcal{A}_\sigma : \xi_{k,1} < -\epsilon\}$$

is at most

$$\left\lfloor \left(\frac{f_0 - f_{min}}{\eta \sigma_{max}^{-3}} \right) \epsilon^{-3} \right\rfloor =: K_{\sigma,2}(\epsilon) \geq 0.$$

Proof. For all $k \in \mathcal{K}_{\epsilon,2}$, we have by (2.1b) that $\lambda_k \geq -\xi_{k,1} > \epsilon$, which along with Steps 5–6 and Lemma 15 implies

$$f_k - f_{k+1} \geq \eta \|s_k\|_2^3 \geq \eta \lambda_k^3 \sigma_k^{-3} \geq \eta \epsilon^3 \sigma_{max}^{-3}.$$

In addition, we have by Theorem 5 that $|\mathcal{K}_{\epsilon,2}| < \infty$. Hence, the reduction in f obtained immediately after the largest index in $\mathcal{K}_{\epsilon,2}$, call it $\bar{k}_{\epsilon,2}$, satisfies

$$f_0 - f_{\bar{k}_{\epsilon,2}+1} = \sum_{k=0}^{\bar{k}_{\epsilon,2}} (f_k - f_{k+1}) \geq \sum_{k \in \mathcal{K}_{\epsilon,2}} (f_k - f_{k+1}) \geq |\mathcal{K}_{\epsilon,2}| \eta \epsilon^3 \sigma_{max}^{-3}.$$

Rearranging this inequality to yield an upper bound for $|\mathcal{K}_{\epsilon,2}|$ and using the fact that $f_0 - f_{min} \geq f_0 - f_{\bar{k}_{\epsilon,2}+1}$, we obtain the desired result. □

We now present our main complexity result of this subsection.

Theorem 6. *Under Assumptions 3, 4, and 5, for a scalar $\epsilon \in (0, \infty)$, the total number of elements in the index set*

$$\{k \in \mathbb{N}_+ : \|g_k\|_2 > \epsilon \text{ or } \xi_{k,1} < -\epsilon\}$$

is at most

$$K_2(\epsilon) := 1 + (K_\sigma(\epsilon) + K_{\sigma,2}(\epsilon) + K_\Delta)(1 + K_\mathcal{C}), \tag{2.28}$$

where $K_\sigma(\epsilon)$, K_Δ, $K_\mathcal{C}$, and $K_{\sigma,2}(\epsilon)$ are defined in Lemmas 17, 18, 21, and 22, respectively. Consequently, we have that $K_2(\epsilon) = \mathcal{O}(\epsilon^{-3})$ as $\epsilon \searrow 0$.

Proof. The proof follows in a similar manner as that of Theorem 4, except that we must now account for additional accepted steps—an upper bound for which is provided by Lemma 22—that may occur until the leftmost eigenvalue of the Hessian matrix is above the desired threshold. □

We remark that the specific upper bound in (2.28) may be reduced by observing that the terms $K_\sigma(\epsilon) + K_{\sigma,2}(\epsilon)$ might double-count certain accepted steps. However, we have presented the result in the above manner for simplicity.

2.2.4 Local convergence to a strict local minimizer

Our goal in this subsection is to prove that if there exists a limit point of $\{x_k\}$ at which the second-order sufficient condition for optimality is satisfied, then, in fact, the entire sequence of iterates converges to this point, and the asymptotic rate of convergence is Q-quadratic. For this purpose, in addition to Assumptions 3 and 4, we make the following assumption. (Note that our analysis in this subsection does not require Assumption 5; instead, the following assumption only involves the looser requirement that the Hessian function of f is locally Lipschitz in a neighborhood about a particular limit point of the iterate sequence.)

Assumption 6. *For some infinite $\mathcal{S} \subseteq \mathbb{N}_+$, the subsequence of iterates $\{x_k\}_{k\in\mathcal{S}}$ converges to $x_* \in \mathbb{R}^n$ at which the Hessian of f is positive definite, i.e., $H(x_*) \succ 0$. Furthermore, there exists a nonempty open convex set about x_* in which the Hessian function H is locally Lipschitz continuous with a scalar Lipschitz constant $H_{Loc} > 0$.*

Our first result, the proof of which follows as for a traditional trust region algorithm, states that the entire iterate sequence converges.

Lemma 23. *The sequence $\{x_k\}$ converges to x_* and $g(x_*) = 0$.*

Proof. By Theorem 3, it follows that $g(x_*) = 0$. The remainder of the result follows similarly to that of [23, Theorem 6.5.2]. □

The next lemma reveals asymptotic properties of the computed trial steps.

Lemma 24. *There exists an iteration number $k_{\mathcal{A}} \in \mathbb{N}_+$ such that, for all $k \in \mathbb{N}_+$ with $k \geq k_{\mathcal{A}}$, the trial step, dual variable, and iteration number satisfy $s_k = -H_k^{-1}g_k$, $\lambda_k = 0$, and $k \in \mathcal{A}$, respectively. That is, eventually, all computed trial steps are Newton steps that are accepted by the algorithm.*

Proof. By Lemma 23, continuity of the Hessian function implies that $H(x_k) \succ 0$ for all sufficiently large $k \in \mathbb{N}_+$. Hence, for all such k, we either have $s_k = -H_k^{-1}g_k$ or the Newton step $-H_k^{-1}g_k$ lies outside the trust region, i.e., in either case,

$$\|s_k\|_2 \leq \|H_k^{-1}g_k\|_2 \leq \|H_k^{-1}\|_2\|g_k\|_2 \implies \|g_k\|_2 \geq \|s_k\|_2/\|H_k^{-1}\|_2. \tag{2.29}$$

Along with Lemma 2 (specifically, (2.8)), this implies that

$$\begin{aligned} f_k - q_k(s_k) &\geq \tfrac{1}{2}\|g_k\|_2 \min\left\{\delta_k, \frac{\|g_k\|_2}{\|H_k\|_2}\right\} \\ &\geq \tfrac{1}{2}\frac{\|s_k\|_2}{\|H_k^{-1}\|_2}\min\left\{\|s_k\|_2, \frac{\|s_k\|_2}{\|H_k\|_2\|H_k^{-1}\|_2}\right\} \\ &\geq \tfrac{1}{2}\frac{\|s_k\|_2^2}{\|H_k\|_2\|H_k^{-1}\|_2^2}. \end{aligned}$$

Thus, with $\xi_* := 1/(4\|H(x_*)\|_2\|H(x_*)^{-1}\|_2^2)$, we have for sufficiently large $k \in \mathbb{N}_+$

$$f_k - q_k(s_k) \geq \xi_*\|s_k\|_2^2.$$

Furthermore, since $\{x_k\} \to x_*$ implies $\{H_k^{-1}g_k\} \to 0$, it follows from (2.29) that $\{s_k\} \to 0$. Combining these facts with the above displayed inequality, we have, as in the proof of Lemma 13, that, for sufficiently large $k \in \mathbb{N}_+$,

$$\begin{aligned} f_k - f(x_k + s_k) &= f_k - q_k(s_k) + q_k(s_k) - f(x_k + s_k) \\ &\geq \xi_*\|s_k\|_2^2 - \tfrac{1}{2}H_{Loc}\|s_k\|_2^3 \geq \eta\|s_k\|_2^3, \end{aligned}$$

meaning that for all sufficiently large $k \in \mathbb{N}_+$ we have $\rho_k \geq \eta$ so that $k \in \mathcal{A} \cup \mathcal{E}$.

By the result of the previous paragraph and Lemma 5, it follows that there exists a scalar constant $\delta_{min} > 0$ such that $\delta_k \geq \delta_{min}$ for all sufficiently large $k \in \mathbb{N}_+$. Moreover, continuity of the gradient and Hessian functions imply that for some sufficiently small $\tau \in \mathbb{R}_{++}$ we have

$$\|H_k^{-1}g_k\|_2 < \delta_{min} \quad \text{whenever} \quad \|x_k - x_*\|_2 \leq \tau.$$

Hence, for sufficiently large $k \in \mathbb{N}_+$, we have $H_k \succ 0$ and the Newton step lies in the trust region, which implies that $s_k = -H_k^{-1}g_k$ and $\lambda_k = 0$ for all sufficiently large $k \in \mathbb{N}_+$. Since we have shown that $k \in \mathcal{A} \cup \mathcal{E}$ for all sufficiently large $k \in \mathbb{N}_+$, this implies that, in fact, $k \in \mathcal{A}$ for all sufficiently large $k \in \mathbb{N}_+$, as desired. □

We now provide our main local convergence rate result.

Theorem 7. *Under Assumptions 3, 4, and 6, it holds that*

$$\|g_{k+1}\|_2 = \mathcal{O}(\|g_k\|_2^2) \quad \textit{and} \quad \|x_{k+1} - x_*\|_2 = \mathcal{O}(\|x_k - x_*\|_2^2) \tag{2.30}$$

i.e., the sequences $\{g_k\}$ *and* $\{x_k\}$ *Q-quadratically converge to* 0 *and* x_**, respectively.*

Proof. With the conclusions of Lemmas 23 and 24, the result follows from standard theory of Newton's method for unconstrained optimization; e.g., see [31]. □

2.3 Numerical Experiments

We implemented the proposed algorithm along with the traditional trust region (TTR) and a standard ARC algorithm in MATLAB for a preliminary performance examination. Figure 2.1 compares the performance of the implemented algorithms using performance profiles (see [32]) for the number of iterations, function/gradient evaluations, and matrix factorizations. The results show that TRACE is at least competitive in terms of number of iterations and function, gradient, and Hessian evaluations, while is completely superior in terms of matrix decompositions. One can justify this superiority of TRACE by noticing that there are some iterations in which TRACE only needs to solve a system of linear equations with just one matrix factorization. We admit that this result is based on our preliminary implementation of these algorithms and restricted to the unconstrained problems in the `CUTEr` test collection [40]. However, we believe that the result from a larger scale experiment would not be very different.

For practical purposes, in the implementation of TRACE we have changed the definition of ρ_k to

$$\rho_k := \frac{f_k - f(x_k + s_k)}{\min\{\|s_k\|_2^3, f_k - m_k^c(s_k; \underline{\sigma})\}},$$

where

$$m_k^c(\sigma_k) := f_k + g_k^T s_k + \frac{1}{2} s_k^T H_k s_k + \frac{\sigma}{3}\|s_k\|_2^3.$$

This change guarantees that the function decrease for a successful step s_k is proportional to $\|s_k\|_2^3$ as it is proved in [14]. In addition, this change relaxes the acceptance condition when the norm of s_k is large. At these points, using the modified ρ_k may improve the performance of the algorithm because it will decrease the number of unsuccessful steps.

To run this experiment, we removed 9 problems due to memory or decoding errors. In addition, 21 problems on which all algorithms failed were removed. The performance profiles shown in Figure 2.1 are based on 130 remaining unconstrained problems in the `CUTEr` collection. For all of these algorithms, the same subproblem solver [23] was used to

find an approximate global solution of the subproblem. We terminated the algorithms as soon as they reached

$$\|g_k\|_\infty \leq 10^{-6} \max\{\|g_0\|_\infty, 1\}.$$

Updating the parameters was done as

$$\text{(TTR)} \qquad \delta_{k+1} \leftarrow \begin{cases} \max\{\delta_k, 2\|s_k\|_2\} & \text{if } \rho_k^q \geq \eta_2 \\ \delta_k & \text{if } \rho_k^q \in [\eta_1, \eta_2) \\ \delta_k/2 & \text{if } \rho_k^q < \eta_1 \end{cases},$$

$$\text{(ARC)} \qquad \sigma_{k+1} \leftarrow \begin{cases} \sigma_k/2 & \text{if } \rho_k^c \geq \eta_2 \\ \sigma_k & \text{if } \rho_k^c \in [\eta_1, \eta_2) \\ 2\sigma_k & \text{if } \rho_k^c < \eta_1 \end{cases}, \text{ and}$$

$$\text{(TRACE)} \qquad \delta_{k+1} \leftarrow \begin{cases} \max\{\delta_k, 2\|s_k\|_2\} & \text{if } \rho_k \geq \eta_2 \\ \delta_k & \text{if } \rho_k \in [\eta_1, \eta_2) \\ \text{CONTRACT} & \text{if } \rho_k < \eta_1 \end{cases},$$

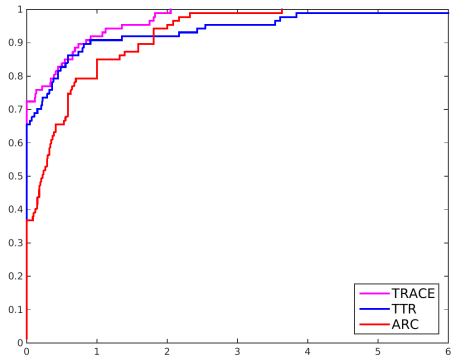
where CONTRACT (see Algorithm 6) uses

$$\underline{\sigma} = 10^{-10}, \quad \overline{\sigma} = 10^{10}, \quad \gamma_\lambda = 2, \quad \gamma_c = 10^{-2}.$$

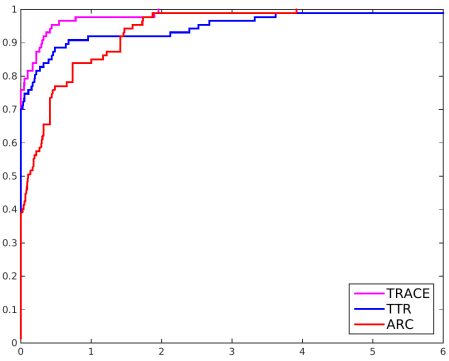
We also collected some extra statistics for TRACE to have a better understanding of the performance of the algorithm. According to our experiment, only 1.01% of the iterations were expansions (Table 2.1). In addition, there was no contraction through Step 31 (Table 2.2). This observation shows that although defining a contraction step as the one in Step 31 is necessary for the analysis, using this type of contraction in practice is very rare. In fact, appropriate choices of $\underline{\sigma}$ and $\overline{\sigma}$ can decrease the usage of Step 31.

Table 2.1: Percentage of iteration types

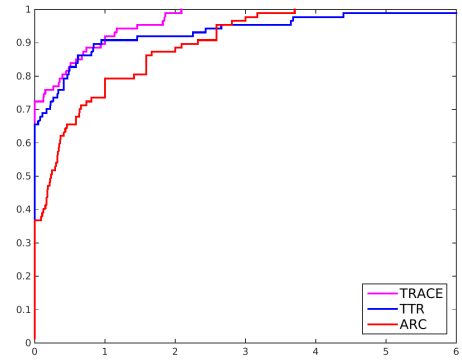
Accepted	Contraction	Expansion
63.73%	35.26%	1.01%



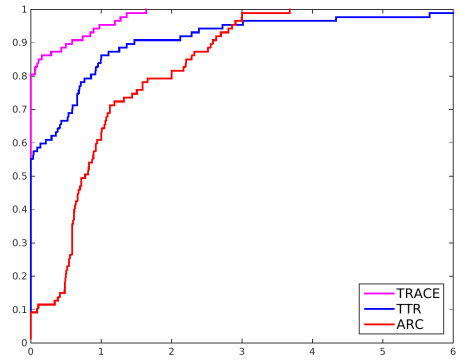
(a) Performance profile for the number of function evaluations



(b) Performance profile for the number of gradient evaluations



(c) Performance profile for the number of iterations



(d) Performance profile for the number of matrix factorizations

Figure 2.1: Performance profiles comparing numbers of evaluations, iterations, and matrix factorizations between TRACE, TTR, and ARC.

Table 2.2: Percentage of contraction types

$\lambda_k + (\underline{\sigma}\|g_k\|_2)^{1/2}$	$\underline{\sigma} \leq \lambda/\|s\|_2 \leq \overline{\sigma}$	$\gamma_\lambda \lambda_k$	$\delta \leftarrow \gamma_c \|s_k\|_2$
2.70%	0.00%	88.09%	9.21%

Chapter 3

An Inexact Regularized Newton Framework with a Worst-Case Iteration Complexity of $\mathcal{O}(\epsilon^{-3/2})$ for Nonconvex Optimization

In this chapter[1], we proposes another algorithm for solving problem (1.1), where the (possibly nonconvex) objective function $f : \mathbb{R}^n \to \mathbb{R}$ is assumed to be twice-continuously differentiable. The optimization problem (1.1) has been widely studied, as evidenced by its appearance as the focal point of numerous textbooks; e.g., see [1], [3], [23], [46], [57], and [60].

For many years, the most popular methods for solving (1.1) were in classes known as line search and trust region methods. Recently, however, cubic regularization methods have become popular, which are based on the pioneering work by [45] and [56]. Their rise in popularity is due to increased interest in algorithms with improved complexity properties, which stems from the impact of so-called optimal algorithms for solving convex optimization problems. For problem (1.1), by complexity properties, we mean a guaranteed bound on the number of iterations (or function evaluations or derivative evaluations) needed by an algorithm before the norm of the gradient of the objective must fall below a

[1] A paper containing the original material of this chapter has been accepted for publication in the *IMA Journal of Numerical Analysis*. Please refer to [30].

positive threshold $\epsilon > 0$. The complexity of a traditional trust region method (e.g., see Algorithm 6.1.1 in [23]) is $\mathcal{O}(\epsilon^{-2})$ (see [13] and Chapter 1), which falls short of the $\mathcal{O}(\epsilon^{-3/2})$ complexity for cubic regularization methods (e.g., see the ARC method by [14, 15]). This latter complexity is optimal among a certain broad class of second-order methods when employed to minimize a broad class of objective functions; see [16]. That said, one can obtain even better complexity properties if higher-order derivatives are used; see [8] and [21].

The better complexity properties of regularization methods such as ARC have been a major point of motivation for discovering other methods that attain the same worst-case iteration complexity bounds. For example, the recently introduced (nontraditional) trust region method known as TRACE (see [28] and Chapter 2) has the same optimal $\mathcal{O}(\epsilon^{-3/2})$ complexity, while at the same time allowing traditional trust region trial steps to be computed and used. A key aspect of the TRACE framework is that a solution to an implicit trust region problem is obtained by varying a regularization parameter instead of a trust region radius. This key idea has been adopted and advanced further by [5]; in particular, they propose an algorithm that has optimal iteration complexity by solving quadratic subproblems that have a carefully chosen quadratic regularization parameter.

The main contributions of this chapter relate to advancing the understanding of optimal complexity algorithms for solving the smooth optimization problem (1.1). Our proposed framework is intentionally very general; it is not a trust region method, a quadratic regularization method, or a cubic regularization method. Rather, we propose a generic set of conditions that each trial step must satisfy that still allow us to establish an optimal first-order complexity result as well as a second-order complexity bound similar to the methods above. Our framework contains as special cases other optimal complexity algorithms such as ARC and TRACE. To highlight this generality of our contribution, we describe one particular instance of our framework that appears to be new to the literature.

During the final preparation of the material of this chapter, we came across the work in [33] and [34]. This work shares certain commonalities with our own and appears to have been developed at the same time. Although there are numerous differences, we shall only point out three of them. First, the precise conditions that they require for each trial step are different from ours. In particular, the condition stated as (3.1c) in [34] requires that regularization is used to compute every trial step, a property not shared

by our method (which can employ Newton steps). Second, they do not consider second-order convergence or complexity properties, although they might be able to do so by incorporating second-order conditions similar to ours. Third, they focus on strategies for identifying an appropriate value for the regularization parameter. An implementation of our method might consider their proposals, but could employ other strategies as well. In any case, overall, we believe that our work are quite distinct, and in some ways are complementary.

Organization In §3.1, we present our general framework that is formally stated as Algorithm 7. In §3.2, we prove that our framework enjoys first-order convergence (see §3.2.1), an optimal first-order complexity (see §3.2.2), and certain second-order convergence and complexity guarantees (see §3.2.3). In §3.3, we show that ARC and TRACE can be viewed as special cases of our framework, and present yet another instance that is distinct from these methods. In §3.4, we present details of implementations of a cubic regularization method and our newly proposed instance of our framework, and provide the results of numerical experiments with both.

Notation We use $\mathbb{R}_+$ to denote the set of nonnegative scalars, $\mathbb{R}_{++}$ to denote the set of positive scalars, and $\mathbb{N}_+$ to denote the set of nonnegative integers. Given a real symmetric matrix A, we write $A \succeq 0$ (respectively, $A \succ 0$) to indicate that A is positive semidefinite (respectively, positive definite). Given a pair of scalars $(a, b) \in \mathbb{R} \times \mathbb{R}$, we write $a \perp b$ to indicate that $ab = 0$. Similarly, given such a pair, we denote their maximum as $\max\{a, b\}$ and their minimum as $\min\{a, b\}$. Given a vector v, we denote its (Euclidean) ℓ_2-norm as $\|v\|$. Finally, given a discrete set $\mathcal{S}$, we denote its cardinality by $|\mathcal{S}|$.

Corresponding to the objective $f : \mathbb{R}^n \to \mathbb{R}$, we define the gradient function $g := \nabla f : \mathbb{R}^n \to \mathbb{R}^n$ and the Hessian function $H := \nabla^2 f : \mathbb{R}^n \to \mathbb{R}^{n \times n}$. Given an iterate x_k in an algorithm for solving (1.1), we define $f_k := f(x_k)$, $g_k := g(x_k) := \nabla f(x_k)$, and $H_k := H(x_k) := \nabla^2 f(x_k)$. Similarly, we apply a subscript to other algorithmic quantities whose definition depends on the iteration number k.

3.1 Algorithm Description

Our algorithm involves generic conditions that a trial step toward solving problem (1.1) must satisfy. A step satisfying these conditions can be obtained by computing—for appropriate positive lower and upper bounds σ_k^{L} and σ_k^{U}, respectively, on the ratio between a regularization variable $\lambda \geq 0$ and the norm of the trial step—an approximate solution of the subproblem

$$\begin{aligned} \mathcal{P}_k(\sigma_k^{\mathrm{L}}, \sigma_k^{\mathrm{U}}) \;:\; \min_{(s,\lambda)\in\mathbb{R}^n\times\mathbb{R}_+} \; & f_k + g_k^T s + \tfrac{1}{2} s^T (H_k + \lambda I) s \\ \text{s.t. } & (\sigma_k^{\mathrm{L}})^2 \|s\|^2 \leq \lambda^2 \leq (\sigma_k^{\mathrm{U}})^2 \|s\|^2. \end{aligned} \tag{3.1}$$

For a given value of the regularization variable λ, this problem involves a quadratic objective function and an upper bound on the norm of the trial step, just as in a trust region method. However, it also includes a lower bound on the norm of the trial step, and, in general, with λ as a variable, it encapsulates other types of subproblems as well, including those present in a cubic regularization framework. For additional details on the properties of this subproblem and its solutions, see Appendices B.1 and B.2.

The conditions that the kth trial step and regularization pair, i.e., (s_k, λ_k), must satisfy are stated in Assumption 7 below, wherein we invoke the following (unregularized) quadratic model of f at x_k:

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

Assumption 7. *The pair (s_k, λ_k) is computed such that it is feasible for problem (3.1) and, with*

$$\Delta_k(s_k, \lambda_k) := \begin{cases} \|s_k\| & \text{if } \lambda_k = 0 \\ \frac{1}{\sqrt{6}} \sqrt{\frac{\|g_k\|\|s_k\|}{\lambda_k}} & \text{if } \lambda_k > 0 \end{cases} \tag{3.2}$$

and constants $(\kappa_1, \kappa_2, \kappa_3) \in \mathbb{R}_{++} \times \mathbb{R}_{++} \times \mathbb{R}_{++}$, the following hold:

$$f_k - q_k(s_k) \geq \frac{\|g_k\|}{6\sqrt{2}} \min\left\{ \frac{\|g_k\|}{1 + \|H_k\|}, \Delta_k(s_k, \lambda_k) \right\}; \tag{3.3a}$$

$$s_k^T (g_k + (H_k + \lambda_k I) s_k) \leq \min\{\kappa_1 \|s_k\|^2, \tfrac{1}{2} s_k^T (H_k + \lambda_k I) s_k + \tfrac{1}{2} \kappa_2 \|s_k\|^3\}; \quad \textit{and} \tag{3.3b}$$

$$\|g_k + (H_k + \lambda_k I) s_k\| \leq \lambda_k \|s_k\| + \kappa_3 \|s_k\|^2. \tag{3.3c}$$

To see that Assumption 7 is well-posed and consistent with problem (3.1), we refer the reader to Theorem 17 in Appendix B.2 wherein we prove that any solution of problem (3.1) with s restricted to a sufficiently large dimensional subspace of $\mathbb{R}^n$ satisfies all of the conditions in Assumption 7. We also claim that one can obtain a pair satisfying Assumption 7 in either of the following two ways:

- Choose $\sigma \in [\sigma_k^{\text{L}}, \sigma_k^{\text{U}}]$, compute s_k by minimizing the cubic function

$$c_k(s;\sigma) := q_k(s) + \tfrac{1}{2}\sigma\|s\|^3 = f_k + g_k^T s + \tfrac{1}{2}s^T H_k s + \tfrac{1}{2}\sigma\|s\|^3 \tag{3.4}$$

  over a sufficiently large dimensional subspace of $\mathbb{R}^n$ (assuming, when $\sigma = \sigma_k^{\text{L}} = 0$, that this function is not unbounded below), then set $\lambda_k \leftarrow \sigma\|s_k\|$. This is essentially the strategy employed in cubic regularization methods such as ARC.

- Choose $\lambda_k \geq 0$, then compute s_k by minimizing the objective of (3.1) with $\lambda = \lambda_k$ over a sufficiently large dimensional subspace of $\mathbb{R}^n$ (assuming that the function is not unbounded below). The resulting pair (s_k, λ_k) satisfies Assumption 7 as long as it is feasible for (3.1). This is essentially the strategy employed in [5] and partly employed in TRACE.

One can imagine other approaches as well. Overall, we state problem (3.1) as a guide for various techniques for computing the pair (s_k, λ_k). Our theory simply relies on the fact that any such computed pair satisfies the conditions in Assumption 7.

Our algorithm, stated as Algorithm 7, employs the following ratio (also employed, e.g., in TRACE) to determine whether a given trial step is accepted or rejected:

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

One potential drawback of employing this ratio is that it is not invariant to scaling of the objective function. However, this choice can be justified. For example, if one were to compute s_k by minimizing the cubic model (3.4) for some $\sigma > 0$, then the reduction in this model yielded by s_k is bounded below by a fraction of $\sigma\|s_k\|^3$ (see [15, Lemma 4.2]), meaning that $\rho_k \geq \eta$ holds when $\sigma \geq \eta$ and the actual reduction in f is proportional to the reduction in the cubic model. For further justification for this choice—such as how it allows the algorithm to accept Newton steps when $\|s_k\|$ is small—we refer the reader to

[5] and Chapter 2 of this dissertation.

Algorithm 7 Inexact Regularized Newton Framework

Require: an acceptance constant $\eta \in \mathbb{R}_{++}$ with $0 < \eta < 1$
Require: bound update constants $\{\gamma_1, \gamma_2\} \subset \mathbb{R}_{++}$ with $1 < \gamma_1 \leq \gamma_2$
Require: ratio lower and upper bound constants $\{\underline{\sigma}, \overline{\sigma}\} \subset \mathbb{R}_{++}$ such that $\overline{\sigma} \geq \underline{\sigma}$

procedure Inexact Regularized Newton
  set $x_0 \in \mathbb{R}^n$
  set $\sigma_0^{\mathrm{L}} \leftarrow 0$ and $\sigma_0^{\mathrm{U}} \in [\underline{\sigma}, \overline{\sigma}]$
  for $k \in \mathbb{N}_+$ **do**
    set (s_k, λ_k) satisfying Assumption 7
    if $\rho_k \geq \eta$ **then** [accept step]
      set $x_{k+1} \leftarrow x_k + s_k$
      set $\sigma_{k+1}^{\mathrm{L}} \leftarrow 0$ and $\sigma_{k+1}^{\mathrm{U}} \leftarrow \sigma_k^{\mathrm{U}}$
    else (i.e., $\rho_k < \eta$) [reject step]
      set $x_{k+1} \leftarrow x_k$
      if $\lambda_k < \underline{\sigma}\|s_k\|$ **then**
        set $\sigma_{k+1}^{\mathrm{L}} \in [\underline{\sigma}, \overline{\sigma}]$ and $\sigma_{k+1}^{\mathrm{U}} \in [\sigma_{k+1}^{\mathrm{L}}, \overline{\sigma}]$
      else
        set $\sigma_{k+1}^{\mathrm{L}} \leftarrow \gamma_1 \frac{\lambda_k}{\|s_k\|}$ and $\sigma_{k+1}^{\mathrm{U}} \leftarrow \gamma_2 \frac{\lambda_k}{\|s_k\|}$

3.2 Convergence Analysis

In this section, we prove global convergence guarantees for Algorithm 7. In particular, we prove under common assumptions that, from remote starting points, the algorithm converges to first-order stationarity, has a worst-case iteration complexity to approximate first-order stationarity that is on par with the methods in Chapter 2, [15], and [5], and—at least in a subspace determined by the search path of the algorithm—converges to second-order stationarity with a complexity on par with the methods in Chapter 2 and [15].

3.2.1 First-order global convergence

Our goal in this subsection is to prove that the sequence of objective gradients vanishes. We make the following assumption about the objective function, which is assumed to hold throughout this section.

Assumption 8. *The objective function $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable and bounded below by a scalar $f_{\inf} \in \mathbb{R}$ on $\mathbb{R}^n$.*

We also make the following assumption related to the sequence of iterates.

Assumption 9. *The gradient function $g : \mathbb{R}^n \to \mathbb{R}^n$ is Lipschitz continuous with Lipschitz constant $g_{Lip} \in \mathbb{R}_{++}$ in an open convex set containing the sequences $\{x_k\}$ and $\{x_k + s_k\}$. Furthermore, the gradient sequence $\{g_k\}$ has $g_k \neq 0$ for all $k \in \mathbb{N}_+$ and is bounded in that there exists a scalar constant $g_{max} \in \mathbb{R}_{++}$ such that $\|g_k\| \leq g_{max}$ for all $k \in \mathbb{N}_+$.*

It is worthwhile to note in passing that our complexity bounds for first- and second-order stationarity remain true even if one were to consider the possibility that $g_k = 0$ for some $k \in \mathbb{N}_+$, in which case one would have the algorithm terminate finitely or, if $H_k \not\succeq 0$, compute an improving direction of negative curvature for H_k. However, allowing this possibility—which is typically unlikely ever to occur in practice—would only serve to obscure certain aspects of our analysis. We refer the reader, e.g., to [15] (specifically, to the discussions at the ends of §2.1, §4, and §5 in that work) for commentary about why zero gradient values do not ruin complexity guarantees such as we present.

We begin with two lemmas each revealing an important consequence of Assumptions 8 and 9.

Lemma 25. *For all $k \in \mathbb{N}_+$, it follows that $s_k \neq 0$.*

Proof. The result follows by combining that $g_k \neq 0$ for all $k \in \mathbb{N}_+$ (see Assumption 9) with (3.3c). □

Lemma 26. *The Hessian sequence $\{H_k\}$ is bounded in norm in that there exists a scalar constant $H_{max} \in \mathbb{R}_{++}$ such that $\|H_k\| \leq H_{max}$ for all $k \in \mathbb{N}_+$.*

Proof. The result follows by Assumption 8, the Lipschitz continuity of g in Assumption 9, and Lemma 1.2.2 in [54]. □

In our next lemma, we prove an upper bound for the regularization variable λ_k.

Lemma 27. *For all $k \in \mathbb{N}_+$, the pair (s_k, λ_k) satisfies*

$$\lambda_k \leq 2\frac{\|g_k\|}{\|s_k\|} + \tfrac{3}{2}H_{max} + \kappa_1.$$

Proof. Since (3.3a) ensures $q_k(s_k) - f_k \leq 0$, it follows with (3.3b) and Lemma 26 that

$$\begin{aligned} 0 \geq q_k(s_k) - f_k &= g_k^T s_k + \tfrac{1}{2} s_k^T H_k s_k \\ &\geq g_k^T s_k + \tfrac{1}{2} s_k^T H_k s_k + s_k^T (g_k + (H_k + \lambda_k I) s_k) - \kappa_1 \|s_k\|^2 \\ &= 2 g_k^T s_k + \tfrac{3}{2} s_k^T H_k s_k + \lambda_k \|s_k\|^2 - \kappa_1 \|s_k\|^2 \\ &\geq -2 \|g_k\| \|s_k\| - \tfrac{3}{2} H_{max} \|s_k\|^2 + \lambda_k \|s_k\|^2 - \kappa_1 \|s_k\|^2. \end{aligned}$$

After rearrangement and dividing by $\|s_k\|^2 \neq 0$ (see Lemma 25), the desired result follows. □

Using Lemma 27, we now prove a lower bound for the reduction in q_k yielded by s_k.

Lemma 28. *For all $k \in \mathbb{N}_+$, the step s_k satisfies*

$$f_k - q_k(s_k) \geq \frac{\|g_k\|}{6\sqrt{2}} \min \left\{ \frac{\|g_k\|}{1 + H_{max}}, \frac{\|s_k\|}{\sqrt{6}} \sqrt{\frac{\|g_k\|}{2\|g_k\| + \|s_k\|(\frac{3}{2} H_{max} + \kappa_1)}} \right\}.$$

Proof. If $\lambda_k = 0$, then by (3.3a) and Lemma 26 it follows that

$$f_k - q_k(s_k) \geq \frac{\|g_k\|}{6\sqrt{2}} \min \left\{ \frac{\|g_k\|}{1 + \|H_k\|}, \|s_k\| \right\} \geq \frac{\|g_k\|}{6\sqrt{2}} \min \left\{ \frac{\|g_k\|}{1 + H_{max}}, \|s_k\| \right\}.$$

On the other hand, if $\lambda_k > 0$, then (3.3a), Lemma 26, and Lemma 27 imply that

$$\begin{aligned} f_k - q_k(s_k) &\geq \frac{\|g_k\|}{6\sqrt{2}} \min \left\{ \frac{\|g_k\|}{1 + \|H_k\|}, \frac{1}{\sqrt{6}} \sqrt{\frac{\|g_k\| \|s_k\|}{\lambda_k}} \right\} \\ &\geq \frac{\|g_k\|}{6\sqrt{2}} \min \left\{ \frac{\|g_k\|}{1 + H_{max}}, \frac{\|s_k\|}{\sqrt{6}} \sqrt{\frac{\|g_k\|}{2\|g_k\| + \|s_k\|(\frac{3}{2} H_{max} + \kappa_1)}} \right\}. \end{aligned}$$

Combining the inequalities from these two cases, the desired result follows. □

Going forward, for ease of reference, we respectively define sets of indices corresponding to accepted and rejected steps throughout a run of the algorithm as

$$\mathcal{A} := \{k \in \mathbb{N}_+ : \rho_k \geq \eta\} \quad \text{and} \quad \mathcal{R} := \{k \in \mathbb{N}_+ : \rho_k < \eta\}.$$

We now show that if the algorithm were only to compute rejected steps from some iteration onward, then the sequence $\{\lambda_k / \|s_k\|\}$ diverges to infinity.

Lemma 29. *If $k \in \mathcal{R}$ for all sufficiently large $k \in \mathbb{N}_+$, then $\{\lambda_k/\|s_k\|\} \to \infty$.*

Proof. Without loss of generality, assume that $\mathcal{R} = \mathbb{N}_+$. We now prove that the condition in Step 11 cannot be true more than once. Suppose, in iteration $\hat{k} \in \mathbb{N}_+$, Step 12 is reached, which means that $\lambda_{\hat{k}+1}/\|s_{\hat{k}+1}\| \geq \underline{\sigma}$ since $(s_{\hat{k}+1}, \lambda_{\hat{k}+1})$ is required to be feasible for $\mathcal{P}_{\hat{k}+1}(\sigma^{\mathrm{L}}_{\hat{k}+1}, \sigma^{\mathrm{U}}_{\hat{k}+1})$ in Step 5 where $\sigma^{\mathrm{L}}_{\hat{k}+1} \geq \underline{\sigma}$. Therefore, the condition in Step 11 tests false in iteration $(\hat{k}+1)$. Then, from Step 5, Step 14, and the fact that $\gamma_1 > 1$, it follows that $\{\lambda_k/\|s_k\|\}$ is monotonically increasing for all $k \geq \hat{k}$. Therefore, the condition in Step 11 cannot test true for any $k \geq \hat{k}+1$. Now, to see that the sequence diverges, notice from this fact, Step 5, and Step 14, it follows that for all $k \geq \hat{k}+1$ we have $\lambda_{k+1}/\|s_{k+1}\| \geq \gamma_1(\lambda_k/\|s_k\|)$ where $\gamma_1 > 1$. Thus, $\{\lambda_k/\|s_k\|\} \to \infty$, as claimed. □

We now prove that if the gradients are bounded away from zero and the sequence of ratios $\{\lambda_k/\|s_k\|\}$ diverges, then $\rho_k \geq \eta$ for all sufficiently large $k \in \mathbb{N}_+$, meaning that the steps are accepted.

Lemma 30. *Suppose that $\mathcal{I} \subseteq \mathbb{N}_+$ is an infinite index set such that for $\epsilon \in \mathbb{R}_{++}$ independent of k, one finds that $\|g_k\| \geq \epsilon$ for all $k \in \mathcal{I}$ and $\{\lambda_k/\|s_k\|\}_{k\in\mathcal{I}} \to \infty$. Then, for all sufficiently large $k \in \mathcal{I}$, it follows that $\rho_k \geq \eta$, meaning $k \in \mathcal{A}$.*

Proof. From the Mean Value Theorem, there exists $\bar{x}_k \in [x_k, x_k + s_k]$ such that

$$\begin{aligned} q_k(s_k) - f(x_k + s_k) &= (g_k - g(\bar{x}_k))^T s_k + \tfrac{1}{2} s_k^T H_k s_k \\ &\geq -\|g_k - g(\bar{x}_k)\|\|s_k\| - \tfrac{1}{2}\|H_k\|\|s_k\|^2. \end{aligned} \tag{3.5}$$

From this, Lemma 28, and Assumption 9, it follows that, for all $k \in \mathcal{I}$,

$$\begin{aligned} f_k - f(x_k + s_k) &= f_k - q_k(s_k) + q_k(s_k) - f(x_k + s_k) \\ &\geq \frac{\|g_k\|}{6\sqrt{2}} \min\left\{ \frac{\|g_k\|}{1 + H_{max}}, \frac{\|s_k\|}{\sqrt{6}} \sqrt{\frac{\|g_k\|}{2\|g_k\| + \|s_k\|(\frac{3}{2}H_{max} + \kappa_1)}} \right\} \\ &\quad - (g_{Lip} + \tfrac{1}{2}H_{max})\|s_k\|^2 \\ &\geq \frac{\epsilon}{6\sqrt{2}} \min\left\{ \frac{\epsilon}{1 + H_{max}}, \frac{\|s_k\|}{\sqrt{6}} \sqrt{\frac{\epsilon}{2g_{max} + \|s_k\|(\frac{3}{2}H_{max} + \kappa_1)}} \right\} \\ &\quad - (g_{Lip} + \tfrac{1}{2}H_{max})\|s_k\|^2. \end{aligned}$$

This shows that there exists a threshold $s_{thresh} > 0$ such that

$$f_k - f(x_k + s_k) \geq \eta \|s_k\|^3 \quad \text{whenever} \quad k \in \mathcal{I} \quad \text{and} \quad \|s_k\| \leq s_{thresh}.$$

We now claim that $\{\|s_k\|\}_{k\in\mathcal{I}} \to 0$. To prove this claim, suppose by contradiction that there exists an infinite subsequence $\mathcal{I}_s \subseteq \mathcal{I}$ and scalar $\epsilon_s \in \mathbb{R}_{++}$ such that $\|s_k\| \geq \epsilon_s$ for all $k \in \mathcal{I}_s$. It then follows from the boundedness of $\{\|g_k\|\}$ (see Assumption 9) and Lemma 27 that $\{\lambda_k\}_{k\in\mathcal{I}_s}$ is bounded. This allows us to conclude that $\{\lambda_k/\|s_k\|\}_{k\in\mathcal{I}_s}$ is bounded, which contradicts the assumptions of the lemma. Thus, $\{\|s_k\|\}_{k\in\mathcal{I}} \to 0$. Hence, there exists $k_s \in \mathcal{I}$ such that for all $k \in \mathcal{I}$ with $k \geq k_s$ one finds $\|s_k\| \leq s_{thresh}$. Therefore, for all $k \in \mathcal{I}$ with $k \geq k_s$, it follows that $\rho_k \geq \eta$, as claimed. □

Next, we prove that the algorithm produces infinitely many accepted steps.

Lemma 31. *It holds that* $|\mathcal{A}| = \infty$ *and* $\{s_k\}_{k\in\mathcal{A}} \to 0$.

Proof. To derive a contradiction, suppose that $|\mathcal{A}| < \infty$. This implies that there exists k_0 such that, for all $k \geq k_0$, one has $k \in \mathcal{R}$ and $(x_k, g_k, H_k) = (x_{k_0}, g_{k_0}, H_{k_0})$. From this fact and Assumption 9, it follows that $\|g_k\| \geq \epsilon$ for all $k \geq k_0$ for some $\epsilon \in \mathbb{R}_{++}$. From the fact that $k \in \mathcal{R}$ for all $k \geq k_0$ and Lemma 29, it follows that $\{\lambda_k/\|s_k\|\} \to \infty$. This fact and $\|g_k\| \geq \epsilon$ for all $k \geq k_0$ imply that all the conditions of Lemma 30 are satisfied for $\mathcal{I} := \{k \in \mathbb{N}_+ : k \geq k_0\}$; therefore, Lemma 30 implies that for all sufficiently large $k \in \mathcal{I}$, one finds $\rho_k \geq \eta$ so that $k \in \mathcal{A}$, a contradiction.

To complete the proof, notice that the objective function values are monotonically decreasing. Combining this with the condition in Step 6, the fact that f is bounded below by f_{inf} (see Assumption 8), and $|\mathcal{A}| = \infty$, one deduces that $\{s_k\}_{k\in\mathcal{A}} \to 0$, as claimed. □

We now prove that there exists an infinite subsequence of iterates such that the sequence of gradients computed at those points converges to zero.

Lemma 32. *It holds that*

$$\liminf_{k\in\mathbb{N}_+, k\to\infty} \|g_k\| = 0.$$

Proof. To derive a contradiction, suppose that $\liminf_{k\in\mathbb{N}_+,k\to\infty} \|g_k\| > 0$, which along with the fact that $g_{k+1} = g_k$ for any $k \in \mathbb{N}_+ \setminus \mathcal{A}$ means $\liminf_{k\in\mathcal{A},k\to\infty} \|g_k\| > 0$. Thus,

there exists $\epsilon \in \mathbb{R}_{++}$ such that

$$\|g_k\| \geq \epsilon \quad \text{for all sufficiently large} \quad k \in \mathcal{A}. \tag{3.6}$$

Under (3.6), let us prove that $\{\lambda_k\}_{k\in\mathcal{A}} \to \infty$. To derive a contradiction, suppose there exists an infinite $\mathcal{A}_\lambda \subseteq \mathcal{A}$ such that $\lambda_k \leq \lambda_{max}$ for some $\lambda_{max} \in \mathbb{R}_{++}$. On the other hand, by $\{s_k\}_{k\in\mathcal{A}} \to 0$ (see Lemma 31) and (3.3c), it follows that $\{g_k + (H_k + \lambda_k I)s_k\}_{k\in\mathcal{A}_\lambda} \to 0$. Combining the upper bound on $\{\lambda_k\}_{k\in\mathcal{A}_\lambda}$, the fact that $\{s_k\}_{k\in\mathcal{A}} \to 0$, and $\|H_k\| \leq H_{max}$ (see Lemma 26), it follows that $\{g_k\}_{k\in\mathcal{A}_\lambda} \to 0$, which violates (3.6). Therefore, $\{\lambda_k\}_{k\in\mathcal{A}} \to \infty$.

Our next goal is to prove, still under (3.6), that $k \in \mathcal{A}$ for all sufficiently large $k \in \mathbb{N}_+$. To prove this, our strategy is to show that the sets of iterations involving a rejected step followed by an accepted step are finite. In particular, let us define the index sets

$$\mathcal{R}_1 := \{k \in \mathcal{R} : \text{the condition in Step 11 tests true and } (k+1) \in \mathcal{A}\} \quad \text{and}$$
$$\mathcal{R}_2 := \{k \in \mathcal{R} : \text{the condition in Step 11 tests false and } (k+1) \in \mathcal{A}\}.$$

We aim to prove that these are finite. First, consider $\mathcal{R}_1$. To derive a contradiction, suppose that $|\mathcal{R}_1| = \infty$. By definition, for all $k \in \mathcal{R}_1$, the condition in Step 11 tests true, meaning (s_{k+1}, λ_{k+1}) is found in Step 5 satisfying $\lambda_{k+1}/\|s_{k+1}\| \leq \overline{\sigma}$. On the other hand, since $(k+1) \in \mathcal{A}$ for all $k \in \mathcal{R}_1$, it follows from Lemma 31 that $\{s_{k+1}\}_{k\in\mathcal{R}_1} \to 0$. Combining the conclusions of these last two sentences shows that $\{\lambda_{k+1}\}_{k\in\mathcal{R}_1} \to 0$. However, this contradicts the conclusion of the previous paragraph, which showed that $\{\lambda_k\}_{k\in\mathcal{A}} \to \infty$. Hence, we may conclude that $|\mathcal{R}_1| < \infty$. Now consider $\mathcal{R}_2$. To derive a contradiction, suppose that $|\mathcal{R}_2| = \infty$. The fact that the condition in Step 11 tests false for $k \in \mathcal{R}_2$ implies that (s_{k+1}, λ_{k+1}) is found in Step 5 satisfying $\lambda_{k+1}/\|s_{k+1}\| \leq \gamma_2\lambda_k/\|s_k\|$. However, since $\{s_{k+1}\}_{k\in\mathcal{R}_2} \to 0$ (see Lemma 31) and $\{\lambda_{k+1}\}_{k\in\mathcal{R}_2} \to \infty$ (established in the previous paragraph), it follows that $\{\lambda_{k+1}/\|s_{k+1}\|\}_{k\in\mathcal{R}_2} \to \infty$, which combined with the previously established inequality $\lambda_{k+1}/\|s_{k+1}\| \leq \gamma_2\lambda_k/\|s_k\|$ shows that $\{\lambda_k/\|s_k\|\}_{k\in\mathcal{R}_2} \to \infty$. Therefore, with (3.6), the conditions in Lemma 30 hold for $\mathcal{I} = \mathcal{R}_2$, meaning that, for all sufficiently large $k \in \mathcal{R}_2$, the inequality $\rho_k \geq \eta$ holds. This contradicts the fact that $\mathcal{R}_2 \subseteq \mathcal{R}$; hence, we conclude that $\mathcal{R}_2$ is finite. Since $\mathcal{R}_1$ and $\mathcal{R}_2$ are finite, it follows from the logic of Algorithm 7 that either $k \in \mathcal{A}$ for all sufficiently large k or $k \in \mathcal{R}$ for all

sufficiently large k. By Lemma 31, it follows that $k \in \mathcal{A}$ for all sufficiently large k.

Thus far, we have proved under (3.6) that $\{\lambda_k\}_{k\in\mathcal{A}} \to \infty$ and that $k \in \mathcal{A}$ for all large $k \in \mathbb{N}_+$. From this latter fact, it follows that there exists k_σ such that $\sigma_k^{\mathrm{U}} = \sigma_{k_\sigma}^{\mathrm{U}} \in \mathbb{R}_{++}$ for all $k \geq k_\sigma$. In addition, from Step 5, it follows that for $k \geq k_\sigma$ one finds $\lambda_k/\|s_k\| \leq \sigma_k^{\mathrm{U}} = \sigma_{k_\sigma}^{\mathrm{U}} < \infty$. However, this leads to a contradiction to the facts that $\{\lambda_k\}_{k\in\mathcal{A}} \to \infty$ and $\{s_k\}_{k\in\mathcal{A}} \to 0$ (see Lemma 31). Overall, we have shown that (3.6) cannot be true, which proves the desired result. □

We close with our main global convergence result of this subsection, the proof of which borrows much from that of Theorem 3.

Theorem 8. *Under Assumptions 7, 8, and 9, it follows that*

$$\lim_{k\in\mathbb{N}_+, k\to\infty} \|g_k\| = 0. \tag{3.7}$$

Proof. For the purpose of reaching a contradiction, suppose that (3.7) does not hold. Combining this with the fact that $|\mathcal{A}| = \infty$ (see Lemma 31), it follows that there exists an infinite subsequence $\{t_i\} \subseteq \mathcal{A}$ (indexed over $i \in \mathbb{N}_+$) and a scalar $\epsilon > 0$ such that, for all $i \in \mathbb{N}_+$, one finds $\|g_{t_i}\| \geq 2\epsilon > 0$. Also, the fact that $|\mathcal{A}| = \infty$ and Lemma 32 imply that there exists an infinite subsequence $\{\ell_i\} \subseteq \mathcal{A}$ (indexed over $i \in \mathbb{N}_+$) such that, for all $i \in \mathbb{N}_+$ and $k \in \mathbb{N}_+$ with $t_i \leq k < \ell_i$, one finds

$$\|g_k\| \geq \epsilon \quad \text{and} \quad \|g_{\ell_i}\| < \epsilon. \tag{3.8}$$

Let us now restrict our attention to indices in the infinite index set

$$\mathcal{K} := \{k \in \mathcal{A} : t_i \leq k < \ell_i \text{ for some } i \in \mathbb{N}_+\}.$$

Observe from (3.8) that, for all $k \in \mathcal{K}$, it follows that $\|g_k\| \geq \epsilon$. Also, from the definition of $\mathcal{A}$,

$$f_k - f_{k+1} \geq \eta\|s_k\|^3 \quad \text{for all} \quad k \in \mathcal{K} \subseteq \mathcal{A}. \tag{3.9}$$

Since $\{f_k\}$ is monotonically decreasing and bounded below, one finds that $\{f_k\} \to \underline{f}$ for

some $\underline{f} \in \mathbb{R}$, which when combined with (3.9) shows that

$$\lim_{k \in \mathcal{K}, k \to \infty} \|s_k\| = 0. \tag{3.10}$$

Using this fact, Lemma 28, Assumption 9, and the Mean Value Theorem (as it is used in the proof of Lemma 30 to yield (3.5)), it follows that for all sufficiently large $k \in \mathcal{K}$ one has

$$\begin{aligned} f_k - f_{k+1} &= f_k - q_k(s_k) + q_k(s_k) - f(x_k + s_k) \\ &\geq \frac{\|g_k\|}{6\sqrt{2}} \min\left\{ \frac{\|g_k\|}{1 + H_{max}}, \frac{\|s_k\|}{\sqrt{6}} \sqrt{\frac{\|g_k\|}{2\|g_k\| + \|s_k\|(\frac{3}{2}H_{max} + \kappa_1)}} \right\} \\ &\quad - (g_{Lip} + \tfrac{1}{2}H_{max})\|s_k\|^2 \\ &\geq \frac{\varepsilon}{6\sqrt{2}} \frac{\|s_k\|}{\sqrt{6}} \sqrt{\frac{\|g_k\|}{2\|g_k\| + \|s_k\|(\frac{3}{2}H_{max} + \kappa_1)}} - (g_{Lip} + \tfrac{1}{2}H_{max})\|s_k\|^2. \end{aligned}$$

It now follows from (3.8) and (3.10) that, as $k \to \infty$ over $k \in \mathcal{K}$, the square root term in the previous inequality converges to $1/\sqrt{2}$. Since the second term in the previous inequality is of order $\|s_k\|^2$, the first term is of order $\|s_k\|$, and $1/\sqrt{2} > 1/\sqrt{3}$, one can thus conclude that $f_k - f_{k+1} \geq \varepsilon\|s_k\|/36$ for all sufficiently large $k \in \mathcal{K}$. Consequently, it follows that for all sufficiently large $i \in \mathbb{N}_+$ one finds

$$\begin{aligned} \|x_{t_i} - x_{\ell_i}\| &\leq \sum_{k \in \mathcal{K}, k = t_i}^{\ell_i - 1} \|x_k - x_{k+1}\| \\ &= \sum_{k \in \mathcal{K}, k = t_i}^{\ell_i - 1} \|s_k\| \leq \sum_{k \in \mathcal{K}, k = t_i}^{\ell_i - 1} \tfrac{36}{\epsilon}(f_k - f_{k+1}) = \tfrac{36}{\epsilon}(f_{t_i} - f_{\ell_i}). \end{aligned}$$

Since $\{f_{t_i} - f_{\ell_i}\} \to 0$ (recall that $\{f_k\} \to \underline{f}$ monotonically) this implies that $\{\|x_{t_i} - x_{\ell_i}\|\} \to 0$, which, in turn, implies that $\{\|g_{t_i} - g_{\ell_i}\|\} \to 0$ because of the continuity of g. However, this is a contradiction since, for any $i \in \mathbb{N}_+$, we have $\|g_{t_i} - g_{\ell_i}\| \geq \epsilon$ by the definitions of $\{t_i\}$ and $\{\ell_i\}$. Overall, we conclude that our initial supposition must be false, implying that (3.7) holds. □

3.2.2 First-order complexity

Our next goal is to prove, with respect to a prescribed positive threshold, a worst-case upper bound on the number of iterations required for our algorithm to reduce the norm of the gradient below the threshold. In this subsection, along with Assumptions 7, 8, and 9, we add the following.

Assumption 10. *The Hessian function H is Lipschitz continuous on a path defined by the sequence of iterates and trial steps; in particular, it is Lipschitz continuous with a scalar Lipschitz constant $H_{Lip} > 0$ on the set $\{x_k + \tau s_k : k \in \mathbb{N}_+, \tau \in [0,1]\}$.*

We begin our analysis in this subsection by providing a lemma that shows that successful steps always result if λ_k is sufficiently large relative to the size of the step.

Lemma 33. *For any $k \in \mathbb{N}_+$, if the pair (s_k, λ_k) satisfies*

$$\lambda_k \geq (H_{Lip} + \kappa_2 + 2\eta)\|s_k\|, \tag{3.11}$$

then $\rho_k \geq \eta$.

Proof. It follows from Assumption 10 and Taylor's expansion with Lagrange remainder that there exists $\bar{x}_k$ on the line segment $[x_k, x_k + s_k]$ such that

$$q_k(s_k) - f(x_k + s_k) = \tfrac{1}{2} s_k^T (H_k - H(\bar{x}_k)) s_k \geq -\tfrac{1}{2} H_{Lip} \|s_k\|^3. \tag{3.12}$$

Also, it follows from (3.3b) that

$$\begin{aligned} f_k - q_k(s_k) &= -g_k^T s_k - \tfrac{1}{2} s_k^T H_k s_k \\ &= -s_k^T (g_k + (H_k + \lambda_k I) s_k) + \tfrac{1}{2}\lambda_k \|s_k\|^2 + \tfrac{1}{2} s_k^T (H_k + \lambda_k I) s_k \\ &\geq -\tfrac{1}{2} s_k^T (H_k + \lambda_k I) s_k - \tfrac{1}{2}\kappa_2 \|s_k\|^3 + \tfrac{1}{2}\lambda_k \|s_k\|^2 + \tfrac{1}{2} s_k^T (H_k + \lambda_k I) s_k \\ &= -\tfrac{1}{2}\kappa_2 \|s_k\|^3 + \tfrac{1}{2}\lambda_k \|s_k\|^2. \end{aligned} \tag{3.13}$$

From (3.12) and (3.13), it follows that

$$\begin{aligned} f_k - f(x_k + s_k) &= f_k - q_k(s_k) + q_k(s_k) - f(x_k + s_k) \\ &\geq \tfrac{1}{2}\lambda_k \|s_k\|^2 - \tfrac{1}{2}\kappa_2 \|s_k\|^3 - \tfrac{1}{2} H_{Lip} \|s_k\|^3, \end{aligned}$$

which together with (3.11) implies that $\rho_k \geq \eta$, as claimed. □

We now prove that the sequence $\{\sigma_k^{\mathrm{U}}\}$ is bounded above.

Lemma 34. *There exists a scalar constant $\sigma_{max} \in \mathbb{R}_{++}$ such that, for all $k \in \mathbb{N}_+$,*

$$\sigma_k^{\mathrm{U}} \leq \sigma_{max}.$$

Proof. Consider any $k \in \mathbb{N}_+$. If s_k is accepted (i.e., $k \in \mathcal{A}$), then $\sigma_{k+1}^{\mathrm{U}} \leftarrow \sigma_k^{\mathrm{U}}$. On the other hand, if s_k is rejected (i.e., $k \in \mathcal{R}$), then it follows from Step 12 and Step 14 that $\sigma_{k+1}^{\mathrm{U}} \leq \max\{\overline{\sigma}, \gamma_2 \lambda_k / \|s_k\|\}$. Moreover, since $k \in \mathcal{R}$, meaning that $\rho_k < \eta$, it follows from Lemma 33 that $\lambda_k / \|s_k\|$ is bounded above by $(H_{Lip} + \kappa_2 + 2\eta)$. Thus, it follows that $\sigma_{k+1}^{\mathrm{U}} \leq \max\{\overline{\sigma}, \gamma_2(H_{Lip} + \kappa_2 + 2\eta)\}$ for all $k \in \mathcal{R}$. Overall, the desired result follows for any $\sigma_{max} \geq \max\{\overline{\sigma}, \gamma_2(H_{Lip} + \kappa_2 + 2\eta)\}$. □

We now establish a lower bound on the norm of any accepted trial step.

Lemma 35. *For all $k \in \mathcal{A}$, it follows that*

$$\|s_k\| \geq \left(\tfrac{1}{2} H_{Lip} + 2\sigma_{max} + \kappa_3\right)^{-1/2} \|g_{k+1}\|^{1/2}.$$

Proof. For all $k \in \mathcal{A}$, it follows that

$$\begin{aligned} \|g_{k+1}\| &\leq \|g_{k+1} - (g_k + (H_k + \lambda_k I)s_k)\| + \|g_k + (H_k + \lambda_k I)s_k\| \\ &\leq \|g_{k+1} - (g_k + H_k s_k)\| + \lambda_k \|s_k\| + \|g_k + (H_k + \lambda_k I)s_k\|. \end{aligned} \tag{3.14}$$

By Taylor's theorem and Assumption 10, the first term on the right-hand side of this inequality satisfies

$$\begin{aligned} \|g_{k+1} - (g_k + H_k s_k)\| &\leq \left\| \int_0^1 (H(x_k + \tau s_k) - H_k) s_k d\tau \right\| \\ &\leq \int_0^1 \|H(x_k + \tau s_k) - H_k\| d\tau \cdot \|s_k\| \\ &\leq \int_0^1 \tau d\tau \cdot H_{Lip} \|s_k\|^2 = \tfrac{1}{2} H_{Lip} \|s_k\|^2. \end{aligned}$$

Combining this with (3.14) and observing Step 5, (3.3c), and Lemma 34, it follows that

$$\begin{aligned}\|g_{k+1}\| &\leq \tfrac{1}{2}H_{Lip}\|s_k\|^2 + 2\frac{\lambda_k}{\|s_k\|}\|s_k\|^2 + \kappa_3\|s_k\|^2 \\ &\leq \tfrac{1}{2}H_{Lip}\|s_k\|^2 + 2\sigma_{max}\|s_k\|^2 + \kappa_3\|s_k\|^2,\end{aligned}$$

which, after rearrangement, completes the proof. □

We are now prepared to prove a worst-case upper bound on the total number of accepted steps that may occur for iterations in which the norm of the gradient of the objective is above a positive threshold.

Lemma 36. *For any $\epsilon \in \mathbb{R}_{++}$, the total number of elements in the index set*

$$\mathcal{K}_\epsilon := \{k \in \mathbb{N}_+ : k \geq 1,\ (k-1) \in \mathcal{A},\ \|g_k\| > \epsilon\}$$

is at most

$$\left\lfloor \left(\frac{f_0 - f_{\inf}}{\eta(\frac{1}{2}H_{Lip} + 2\sigma_{max} + \kappa_3)^{-3/2}} \right) \epsilon^{-3/2} \right\rfloor =: N_{\mathcal{A}}(\epsilon) \geq 0. \tag{3.15}$$

Proof. The proof follows in a similar manner as that of Lemma 17. By Lemma 35, it follows that, for all $k \in \mathcal{K}_\epsilon$, one finds

$$\begin{aligned}f_{k-1} - f_k &\geq \eta\|s_{k-1}\|^3 \\ &\geq \eta(\tfrac{1}{2}H_{Lip} + 2\sigma_{max} + \kappa_3)^{-3/2}\|g_k\|^{3/2} \\ &\geq \eta(\tfrac{1}{2}H_{Lip} + 2\sigma_{max} + \kappa_3)^{-3/2}\epsilon^{3/2}.\end{aligned}$$

In addition, it follows from Theorem 8 that $|\mathcal{K}_\epsilon| < \infty$. Hence, the reduction in f obtained up to the largest index in $\mathcal{K}_\epsilon$, call it $\bar{k}_\epsilon$, satisfies

$$f_0 - f_{\bar{k}_\epsilon} = \sum_{k=1}^{\bar{k}_\epsilon}(f_{k-1} - f_k) \geq \sum_{k\in\mathcal{K}_\epsilon}(f_{k-1} - f_k) \geq |\mathcal{K}_\epsilon|\eta(\tfrac{1}{2}H_{Lip} + 2\sigma_{max} + \kappa_3)^{-3/2}\epsilon^{3/2}.$$

Rearranging this inequality to yield an upper bound for $|\mathcal{K}_\epsilon|$ and using the fact that $f_0 - f_{\inf} \geq f_0 - f_{\bar{k}_\epsilon}$, one obtains the desired result. □

In order to prove a result similar to Lemma 36 for the *total* number of iterations with $\|g_k\| > \epsilon$, we require an upper bound on the total number of trial steps that may be

rejected between accepted steps. To this end, let us define, for a given $\hat{k} \in \mathcal{A} \cup \{0\}$, the iteration number and corresponding set

$$k_{\mathcal{A}}(\hat{k}) := \min\{k \in \mathcal{A} : k > \hat{k}\}$$
$$\text{and} \quad \mathcal{I}(\hat{k}) := \{k \in \mathbb{N}_+ : \hat{k} < k < k_{\mathcal{A}}(\hat{k})\},$$

i.e, we let $k_{\mathcal{A}}(\hat{k})$ be the smallest of all iteration numbers in $\mathcal{A}$ that is strictly larger than $\hat{k}$, and we let $\mathcal{I}(\hat{k})$ be the set of iteration numbers between $\hat{k}$ and $k_{\mathcal{A}}(\hat{k})$.

We now show that the number of rejected steps between the first iteration and the first accepted step, or between consecutive accepted steps, is bounded above.

Lemma 37. *For any $\hat{k} \in \mathcal{A} \cup \{0\}$, it follows that*

$$|\mathcal{I}(\hat{k})| \leq 1 + \left\lfloor \frac{1}{\log(\gamma_1)} \log\left(\frac{\sigma_{max}}{\underline{\sigma}}\right) \right\rfloor =: N_{\mathcal{R}} \geq 0.$$

Proof. The proof follows in a similar manner as for Lemma 21. First, the result holds trivially if $|\mathcal{I}(\hat{k})| = 0$. Thus, we may assume that $|\mathcal{I}(\hat{k})| \geq 1$. Since $(\hat{k}+1) \in \mathcal{R}$ by construction, it follows from Steps 11–14 and Step 5 that $\lambda_{\hat{k}+2}/\|s_{\hat{k}+2}\| \geq \underline{\sigma}$, which, due to the lower bound on $\lambda_{k+1}/\|s_{k+1}\|$ in Step 14 and Step 5, leads to

$$\lambda_{k_{\mathcal{A}}(\hat{k})} \geq \underline{\sigma} \, (\gamma_1)^{k_{\mathcal{A}}(\hat{k})-\hat{k}-2} \, \|s_{k_{\mathcal{A}}(\hat{k})}\|.$$

Combining this with Step 5 and Lemma 34 shows that

$$\sigma_{max} \geq \sigma^{\text{U}}_{k_{\mathcal{A}}(\hat{k})} \geq \lambda_{k_{\mathcal{A}}(\hat{k})}/\|s_{k_{\mathcal{A}}(\hat{k})}\| \geq \underline{\sigma} \, (\gamma_1)^{k_{\mathcal{A}}(\hat{k})-\hat{k}-2}.$$

After rearrangement, it now follows that

$$k_{\mathcal{A}}(\hat{k}) - \hat{k} - 2 \leq \frac{1}{\log(\gamma_1)} \log\left(\frac{\sigma_{max}}{\underline{\sigma}}\right).$$

The desired result follows from this inequality since $|\mathcal{I}(\hat{k})| = k_{\mathcal{A}}(\hat{k}) - \hat{k} - 1$. □

We are now prepared to prove our main complexity result of this subsection.

Theorem 9. *Under Assumptions 7, 8, 9, and 10, for a scalar $\epsilon \in \mathbb{R}_{++}$, the total number*

of elements in the index set $\{k \in \mathbb{N}_+ : \|g_k\| > \epsilon\}$ *is at most*

$$N(\epsilon) := 1 + N_{\mathcal{R}} N_{\mathcal{A}}(\epsilon), \tag{3.16}$$

where $N_{\mathcal{A}}(\epsilon)$ *and* $N_{\mathcal{R}}$ *are defined in Lemmas 36 and 37, respectively. Consequently, for any* $\bar{\epsilon} \in \mathbb{R}_{++}$*, it follows that* $N(\epsilon) = \mathcal{O}(\epsilon^{-3/2})$ *for all* $\epsilon \in (0, \bar{\epsilon}]$.

Proof. Without loss of generality, we may assume that at least one iteration is performed. Lemma 36 guarantees that the total number of elements in the index set $\{k \in \mathcal{A} : k \geq 1,\ \|g_k\| > \epsilon\}$ is at most $N_{\mathcal{A}}(\epsilon)$, where, immediately prior to each of the corresponding accepted steps, Lemma 37 guarantees that at most $N_{\mathcal{R}}$ trial steps are rejected. Accounting for the first iteration, the desired result follows. □

3.2.3 Second-order global convergence and complexity

Our goal in this subsection is to prove results showing that, in some sense, the algorithm converges to second-order stationarity and does so with a worst-case iteration complexity on par with the method in [15] and the one proposed in Chapter 2. In particular, our results show that if the algorithm computes each search direction to satisfy a curvature condition over a subspace, then second-order stationarity is reached in a manner that depends on the subspaces.

In this subsection, we make the following additional assumption about the subproblem solver.

Assumption 11. *For all* $k \in \mathbb{N}_+$*, let* $\mathcal{L}_k \subseteq \mathbb{R}^n$ *denote a subspace with an orthonormal basis formed from the columns of a matrix* R_k*. The step* s_k *satisfies*

$$\xi(R_k^T H_k R_k) \geq -\kappa_4 \|s_k\| \tag{3.17}$$

for some $\kappa_4 \in \mathbb{R}_+$*, where* $\xi(R_k^T H_k R_k)$ *indicates the smallest eigenvalue of* $R_k^T H_k R_k$.

This assumption is reasonable, e.g., in cases when s_k is computed by solving problem 3.1 with the component s restricted to a subspace of $\mathbb{R}^n$. We refer the reader to Theorem 17 for a proof of this fact, which also reveals that this assumption is congruous with Assumption 7.

Under this assumption, we have the following second-order convergence result.

Theorem 10. *Suppose Assumptions 7, 8, 9, 10, and 11 hold. It follows that*

$$\liminf_{k \in \mathcal{A}, k \to \infty} \xi(R_k^T H_k R_k) \geq 0.$$

Proof. The result follows from (3.17) since $\{s_k\}_{k \in \mathcal{A}} \to 0$ (see Lemma 31). □

As a consequence of Theorem 10, if the sequence $\{R_k\}_{k \in \mathcal{A}}$ tends toward full-dimensionality as $k \to \infty$, then any limit point x_* of $\{x_k\}$ must have $H(x_*) \succeq 0$.

Our next goal is to prove a worst-case iteration complexity result for achieving second-order stationarity in a sense similar to that in Theorem 10. Toward this end, we first prove the following lemma, which is similar to Lemma 36.

Lemma 38. *For any $\epsilon \in \mathbb{R}_{++}$, the total number of elements in the index set*

$$\mathcal{K}_{\epsilon,\xi} := \{k \in \mathbb{N}_+ : k \geq 1,\ (k-1) \in \mathcal{A},\ \xi(R_k^T H_k R_k) < -\epsilon\}$$

is at most

$$\left\lfloor \left(\frac{f_0 - f_{\text{inf}}}{\eta \kappa_4^{-3}} \right) \epsilon^{-3} \right\rfloor =: N_{\mathcal{A},\xi}(\epsilon) \geq 0. \tag{3.18}$$

Proof. Under Assumption 11, it follows that, for all $k \in \mathcal{K}_{\epsilon,\xi}$, one finds

$$f_{k-1} - f_k \geq \eta \|s_{k-1}\|^3 \geq \eta \left(\frac{-\xi(R_k^T H_k R_k)}{\kappa_4} \right)^3 \geq \eta \kappa_4^{-3} \epsilon^3.$$

It follows from this inequality, the fact that f is monotonically decreasing over the sequence of iterates, and Assumption 8 that

$$f_0 - f_{\text{inf}} \geq \sum_{k \in \mathcal{K}_{\epsilon,\xi}} (f_{k-1} - f_k) \geq |\mathcal{K}_{\epsilon,\xi}| \eta \kappa_4^{-3} \epsilon^3.$$

Rearranging this inequality to yield an upper bound for $|\mathcal{K}_{\epsilon,\xi}|$ gives the result. □

We close with the following second-order complexity result.

Theorem 11. *Under Assumptions 7, 8, 9, 10, and 11, for any pair of scalars $(\epsilon_1, \epsilon_2) \in \mathbb{R}_{++} \times \mathbb{R}_{++}$, the number of elements in the index set*

$$\{k \in \mathbb{N}_+ : \|g_k\| > \epsilon_1 \ \vee \ \xi(R_k^T H_k R_k) < -\epsilon_2\}$$

is at most

$$N(\epsilon_1, \epsilon_2) := 1 + N_{\mathcal{R}} \max\{N_{\mathcal{A}}(\epsilon_1), N_{\mathcal{A},\xi}(\epsilon_2)\}, \tag{3.19}$$

where $N_{\mathcal{A}}(\cdot)$, $N_{\mathcal{R}}$, and $N_{\mathcal{A},\xi}(\cdot)$ are defined in Lemmas 36, 37, and 38, respectively. Consequently, for any pair of scalars $(\bar{\epsilon}_1, \bar{\epsilon}_2) \in \mathbb{R}_{++} \times \mathbb{R}_{++}$, it follows that

$$N(\epsilon_1, \epsilon_2) = \mathcal{O}(\max\{\epsilon_1^{-3/2}, \epsilon_2^{-3}\}) \quad \textit{for all} \quad (\epsilon_1, \epsilon_2) \in (0, \bar{\epsilon}_1] \times (0, \bar{\epsilon}_2].$$

Proof. The proof follows in a similar manner as that of Theorem 9 by additionally incorporating the bound proved in Lemma 38. □

3.3 Algorithm Instances

Algorithm 7 is a broad framework containing, amongst other algorithms, ARC and TRACE. Indeed, the proposed framework and its supporting analyses cover a wide range of algorithms as long as the pairs in the sequence $\{(s_k, \lambda_k)\}$ satisfy Assumption 7.

In this section, we show that ARC and TRACE are special cases of our proposed framework in that the steps these algorithms accept would also be acceptable for our framework, and that the procedures followed by these methods after a step is rejected are consistent with our framework. We then introduce an instance of our frameowork that is new to the literature. (If desired for the guarantees in §3.2.3, one could also mind whether the elements in the sequence $\{(s_k, \lambda_k)\}$ satisfy Assumption 11. However, for brevity in this section, let us suppose that one is interested only in Assumption 7.)

3.3.1 `ARC` as a special case

The ARC method, which was inspired by the work in [45] and [56], was first proposed and analyzed in [14, 15]. In these papers, various sets of step computation conditions are considered involving exact and inexact subproblem solutions yielding different types of convergence and worst-case complexity guarantees. For our purposes here, we consider the more recent variant of ARC stated and analyzed as "ARp" with $p = 2$ in [8]. (For ease of comparison, we consider this algorithm when their regularization parameter update—see Step 4 in their algorithm—uses $\eta_1 = \eta_2$. Our algorithm is easily extended to employ a two-tier acceptance condition, involving two thresholds η_1 and η_2, as is used in [8] and

[14, 15].)

Suppose that a trial step s_k is computed by this version of ARC. In particular, let us make the reasonable assumption that the subproblem for which s_k is an approximate solution is defined by some regularization value $\sigma_k \in [\sigma_k^{\mathrm{L}}, \sigma_k^{\mathrm{U}}]$ (with $\sigma_k^{\mathrm{L}} \geq \sigma_{min}$ since ARC ensures that $\sigma_k \geq \sigma_{min} \in \mathbb{R}_{++}$ for all $k \in \mathbb{N}$) and that this subproblem is minimized over a subspace $\mathcal{L}_k$ such that $g_k \in \mathcal{L}_k$ (see Appendix B.2). As is shown using a similar argument as in the proof of our Theorem 17(b), one can show under these conditions that (s_k, λ_k) with $\lambda_k = \sigma_k \|s_k\|$ satisfies (3.3a). In addition, considering the algorithm statement in [8], but using our notation, one is required to have

$$g_k^T s_k + \tfrac{1}{2} s_k^T H_k s_k + \lambda_k \|s_k\|^2 < 0 \quad \text{and} \quad \|g_k + (H_k + \lambda_k I) s_k\| \leq \theta \|s_k\|^2 \quad \text{for some} \quad \theta \in \mathbb{R}_{++}.$$

It is easily seen that (s_k, λ_k) satisfying these conditions also satisfies (3.3b)–(3.3c) for any $(\kappa_1, \kappa_2, \kappa_3)$ such that $\kappa_1 \geq \frac{1}{2} H_{max}$ and $\kappa_3 \geq \theta$. Overall, we have shown that a trial step s_k computed by this version of ARC satisfies Assumption 7, meaning that it satisfies the condition in Step 5 in Algorithm 7. If this trial step is accepted by ARC, then this means that $f_k - f(x_k + s_k) \geq \eta_1 (f_k - q_k(s_k))$. Along with [8, Lemma 2.1], this implies that $f_k - f(x_k + s_k) \geq \frac{1}{3} \eta \sigma_k \|s_k\|^3$, meaning that $\rho_k \geq \frac{1}{3} \eta_1 \sigma_{min}$. Hence, this trial step would also be accepted in Algorithm 7 under the assumption that $\eta \in (0, \frac{1}{3} \eta_1 \sigma_{min}]$.

Finally, if a trial step is rejected in this version of ARC, then σ_{k+1} is set to a positive multiple of σ_k. This is consistent with the procedure after a step rejection in Algorithm 7, where it is clear that, with appropriate parameter choices, one would find $\sigma_{k+1} \in [\sigma_{k+1}^{\mathrm{L}}, \sigma_{k+1}^{\mathrm{U}}]$.

3.3.2 `TRACE` as a special case

TRACE is proposed and analyzed in Chapter 2. Our goal in this subsection is to show that, with certain parameter settings, a trial step that is computed and accepted by TRACE could also be the one that is computed and accepted by Algorithm 7, and that the procedures for rejecting a step in TRACE are consistent with those in Algorithm 7. Amongst other procedures, TRACE involves dynamic updates for two sequences, $\{\delta_k\}$ and $\{\Delta_k\}$. The elements of $\{\delta_k\}$ are the trust region radii while $\{\Delta_k\}$ is a monotonically nondecreasing sequence of upper bounds for the trust region radii; consequently, $\|s_k\| \leq \delta_k \leq \Delta_k$ with $\Delta_{k+1} \geq \Delta_k$ for all $k \in \mathbb{N}$. For simplicity, let us assume that the trial steps computed

in TRACE satisfy $\|s_k\| < \Delta_k$ for all $k \in \mathbb{N}$. This is a fair assumption since, as shown in Lemma 10, the manner in which $\{\Delta_k\}$ is set ensures that $\|s_k\| = \Delta_k$ only a finite number of times in any run.

In TRACE, during iteration $k \in \mathbb{N}$, a trust region radius $\delta_k \in \mathbb{R}_{++}$ is given and a trial step s_k and regularization value λ_k are computed satisfying the standard trust region subproblem optimality conditions

$$g_k + (H_k + \lambda_k I)s_k = 0, \quad H_k + \lambda_k I \succeq 0, \quad \text{and} \quad \lambda_k(\delta_k - \|s_k\|) = 0,$$

where $(\lambda_k, \delta_k - \|s_k\|) \geq 0$. By the first of these conditions, the pair (s_k, λ_k) clearly satisfies (3.3b)–(3.3c). In addition, one can use standard trust region theory, in particular related to Cauchy decrease (see [23] or [57]), to show that the pair also satisfies (3.3a). Overall, assuming that the pair $(\sigma_k^{\mathrm{L}}, \sigma_k^{\mathrm{U}})$ is set such that $\lambda_k/\|s_k\| \in [\sigma_k^{\mathrm{L}}, \sigma_k^{\mathrm{U}}]$, it follows that Assumption 7 is satisfied, meaning that TRACE offers the condition in Step 5 in Algorithm 7. If the trial step s_k is subsequently accepted by TRACE, then it would also be accepted by Algorithm 7 since both algorithms use the same step acceptance condition.

Now suppose that a trial step is not accepted in TRACE. This can occur in two circumstances. It can occur if $\rho_k \geq \eta$ while $\lambda_k > \sigma_k\|s_k\|$, in which case the trust region radius is *expanded* and a new subproblem is solved. By the proof of Lemma 6, the solution of this new subproblem yields (in iteration $k+1$ in TRACE) the relationship that $\lambda_{k+1}/\|s_{k+1}\| \leq \sigma_{k+1} = \sigma_k$. Hence, under the same assumption as above that the pair $(\sigma_k^{\mathrm{L}}, \sigma_k^{\mathrm{U}})$ is set such that $\lambda_k/\|s_k\| \in [\sigma_k^{\mathrm{L}}, \sigma_k^{\mathrm{U}}]$, this shows that the procedure in TRACE involving *an expansion of the trust region radius and the computation of the subsequent trial step* yields a trial step that would be offered in a *single iteration* in Algorithm 7. The other circumstance in which a trial step is rejected in TRACE is when $\rho_k < \eta$, in which case the trust region radius is contracted. In this case, one can see that the outcome of the CONTRACT subroutine in TRACE is consistent with Steps 11–14 of Algorithm 7 in the sense that the solution of the subsequent subproblem in TRACE will have $\lambda_{k+1}/\|s_{k+1}\| \in [\underline{\sigma}, \overline{\sigma}]$ (if $\lambda_k < \underline{\sigma}\|s_k\|$) or $\lambda_{k+1}/\|s_{k+1}\|$ within a range defined by positive multiples of $\lambda_k/\|s_k\|$; see Lemmas 14 and 20.

3.3.3 A hybrid algorithm

The primary distinguishing feature of our algorithm instance is the manner in which we compute the pair (s_k, λ_k) in Step 5 of Algorithm 7. Our newly proposed hybrid algorithm considers two cases.

Case 1: $\sigma_k^{\text{L}} > 0$. In this case, we find a pair (s_k, λ_k) by solving problem (B.3) over a sequence of increasingly higher dimensional Krylov subspaces as described in [14] until (3.3) and (3.17) are satisfied. The reason we know that (3.3) and (3.17) will eventually be satisfied can be seen as follows. Solving problem (B.3) over a Krylov subspace is equivalent to solving problem (B.7) with an appropriate choice of R_k as a basis for that Krylov subspace, then setting $s_k = R_k v_k$. Then, it follows from Theorem 16(i) that solving (B.7) is equivalent to solving (B.6), which in turn is equivalent to solving (B.5) in the sense that if $(v_k, \lambda_k, \beta_k^{\text{L}}, \beta_k^{\text{U}}, \beta_k^{\text{N}})$ is a first-order primal-dual solution of problem (B.6), then $(s_k, \lambda_k, \beta_k^{\text{L}}, \beta_k^{\text{U}}, \beta_k^{\text{N}})$ with $s_k = R_k v_k$ is a solution of problem (B.5). Finally, we need only note from Theorem 17 that solutions to problem (B.5) satisfy (3.3a) for all Krylov subspaces $\mathcal{L}_k$ (recall that g_k is contained in all Krylov subspaces), (3.3b) for all Krylov subspaces, (3.3c) if the Krylov subspace $\mathcal{L}_k$ includes enough of the space (in the worst case, $\mathcal{L}_k = \mathbb{R}^n$), and (3.17) for all Krylov subspaces.

Case 2: $\sigma_k^{\text{L}} = 0$. In this case, we begin by applying the linear CG method in an attempt to solve the linear system $H_k s = -g_k$, which iteratively solves

$$\min_{s \in \mathbb{R}^n} \; q_k(s) \tag{3.20}$$

over a sequence of expanding Krylov subspaces. One of two outcomes is possible. First, the CG algorithm may ultimately identify a vector s_k such that (s_k, λ_k) with $\lambda_k = 0$ satisfies (3.3) and (3.17). Second, the CG algorithm may never identify a vector s_k such that (s_k, λ_k) with $\lambda_k = 0$ satisfies (3.3) and (3.17). Indeed, this might occur if CG encounters a direction of negative curvature—in which case we terminate CG immediately—or if CG solves (3.20) accurately or reaches an iteration limit, and yet at least one condition in (3.3)/(3.17) is not satisfied. In such a case, we choose to reset $\sigma_k^{\text{L}} \in (0, \sigma_k^{\text{U}}]$, then solve problem (B.3) over a sequence of expanding Krylov subspaces as described in Case 1. In this manner, we are guaranteed to identify a

pair (s_k, λ_k) satisfying (3.3) and (3.17) as required.

3.4 Implementation and Numerical Results

We implemented two algorithms in `MATLAB`, one following the strategy in §3.3.3 and, for comparison purposes, one following the ARC algorithm in [15] with ideas from [8]. We refer to our implementation of the former as `iR_Newton`, for inexact Regularized Newton, and to our implementation of the latter as `iARC`, for inexact ARC. In this section, we describe our approach for computing the pairs $\{(s_k, \lambda_k)\}$ in `iR_Newton` and `iARC`, as well as other implementation details, and discuss the results of numerical experiments on a standard set of nonlinear optimization test problems.

3.4.1 Implementation details

Let us begin by noting that the implemented algorithms terminate in iteration $k \in \mathbb{N}_+$ if

$$\|g_k\|_\infty \leq 10^{-6} \max\{\|g_0\|_\infty, 1\}.$$

We chose not to employ a termination test based on a second-order stationarity condition. Correspondingly, neither of the algorithms check a second-order condition when computing a trial step; e.g., in `iR_Newton`, we are satisfied with a step satisfying (3.3) and do not check (3.17). In addition, for practical purposes, we set an maximum iteration limit of 10^6, a time limit of four hours, and a minimum step norm limit of 10^{-20}. For reference, the input parameter values we used are given in Table 3.1. We chose these values as ones that worked well on our test set for both implemented algorithms.

Table 3.1: Input parameters for `iARC` and `iR_Newton`

η_1	`1.0e-16`	γ_0	`2.0e-01`	κ_1	`1.0e+00`	$\underline{\sigma}$	`1.0e-10`
η_2	`1.0e-01`	γ_1	`1.0e+01`	κ_2	`1.0e+00`	$\overline{\sigma}$	`1.0e+20`
		γ_2	`2.0e+02`	κ_3	`1.0e+00`		

For both implemented algorithms, we employ a sequence $\{\sigma_k\}$ that is updated dynamically. In `iARC`, this sequence is handled as described in [15], namely,

$$\sigma_{k+1} \leftarrow \begin{cases} \max\{\underline{\sigma}, \gamma_0 \sigma_k\} & \text{if } \frac{f_k - f(x_k + s_k)}{f_k - c_k(s_k;\sigma_k)} \geq \eta_2 \\ \sigma_k & \text{if } \frac{f_k - f(x_k + s_k)}{f_k - c_k(s_k;\sigma_k)} \in [\eta_1, \eta_2) \\ \gamma_1 \sigma_k & \text{if } \frac{f_k - f(x_k + s_k)}{f_k - c_k(s_k;\sigma_k)} < \eta_1 \end{cases}$$

The value σ_k is used in defining $c_k(\cdot;\sigma_k)$ (recall (3.4)) that is minimized approximately to compute the trial step s_k for all $k \in \mathbb{N}_+$. In particular, the implementation iteratively constructs Krylov subspaces of increasing dimension using the Lanczos process, where for each subspace we employ the `RQS` function from the `GALAHAD` software library (see [39] and [41]) to minimize $c_k(\cdot;\sigma_k)$ over the subspace. If the subspace is full-dimensional or the resulting step s_k satisfies

$$\|g_k + (H_k + \sigma_k \|s_k\| I) s_k\| \leq \kappa_3 \|s_k\|^2, \tag{3.21}$$

then it is used as the trial step. Otherwise, the process continues with a larger subspace. We remark that condition (3.21) is more restrictive than our condition (3.3c), but we use it since it is one that has been proposed for cubic regularization methods; e.g., see (2.13) in [8].

One could employ more sophisticated techniques for setting the elements of the sequence $\{\sigma_k\}$ in `iARC` that attempt to reduce the number of rejected steps; e.g., see [42]. Such improvements might aid `iR_Newton` as well. However, for simplicity and to avoid the need for additional parameter tuning, we did not include such enhancements in our implemented algorithms.

As for `iR_Newton`, for consistency between the two implementations, we do not explicitly compute the sequence $\{\lambda_k\}$, but rather employ $\{\sigma_k^{\text{L}} \|s_k\|\}$ in its place. For example, whenever an acceptable step is computed with $\sigma_k^{\text{L}} = 0$, then, as described in **Case 2** in §3.3.3, we effectively use $\lambda_k = 0$. On the other hand, when $\sigma_k^{\text{L}} > 0$, we employ the same iterative approach as used for `iARC` to compute the trial step s_k as an approximate minimizer of $c_k(\cdot;\sigma_k^{\text{L}})$, where in place of λ_k in (3.3) we employ $\sigma_k^{\text{L}} \|s_k\|$. Then, in either case, in the remainder of iteration $k \in \mathbb{N}_+$, specifically for setting σ_{k+1}^{L} and σ_{k+1}^{U}, we use $\sigma_k^{\text{L}} \|s_k\|$

in place of λ_k in Steps 11 and 14. We also define an auxiliary sequence $\{\sigma_k\}$ using the update

$$\sigma_{k+1} \leftarrow \begin{cases} \max\{\underline{\sigma}, \gamma_0\sigma_k\} & \text{if } \rho_k \geq \eta_1 \text{ and } \sigma_k^{\text{L}} > 0 \\ \sigma_k & \text{if } \sigma_k^{\text{L}} = 0 \\ \min\{\gamma_1\sigma_k, \overline{\sigma}\} & \text{if } \rho_k < \eta_1 \text{ and } \sigma_k^{\text{L}} > 0. \end{cases}$$

This update is similar to the one employed for `iARC` with the added assurance that $\{\sigma_k\} \subset [\underline{\sigma}, \overline{\sigma}]$. The elements of this sequence are used in two circumstances. First, if, as described in **Case 2** in §3.3.3, CG fails to produce a trial step s_k satisfying (3.3) (with $\lambda_k = 0$), then we reset $\sigma_k^{\text{L}} \leftarrow \sigma_k$ and revert to the same scheme as above to compute the trial step when $\sigma_k^{\text{L}} > 0$. Second, if a step is rejected and $\sigma_k^{\text{L}} < \underline{\sigma}$ (equivalently, $\lambda_k < \underline{\sigma}\|s_k\|_2$ as in Step 12 in Algorithm 7), then we set $\sigma_{k+1}^{\text{L}} \leftarrow \sigma_{k+1}$. Lastly, we note that if CG ever performs n iterations and the resulting solution (due to numerical error) does not satisfy (3.3) and no negative curvature is detected, then the resulting approximate solution s_k is used as the trial step.

3.4.2 Results on the `CUTEst` test set

We employed our implemented algorithms, `iARC` and `iR_Newton`, to solve unconstrained problems in the `CUTEst` test set; see [43]. Among 171 unconstrained problems in the set, one (`FLETCBV2`) was removed since the algorithms terminated at the initial point, five (`ARGLINC`, `DECONVU`, `FLETCHBV`, `INDEFM`, and `POWER`) were removed due to a function evaluation error or our memory limitation of 8GB, and nine (`EIGENBLS`, `EIGENCLS`, `FMINSURF`, `NONMSQRT`, `SBRYBND`, `SCURLY10`, `SCURLY20`, `SCURLY30`, and `SSCOSINE`) were removed since neither algorithm terminated within our time limit. In addition, four were removed since neither of the algorithms terminated successfully: for `HIELOW`, `iARC` reached our maximum iteration limit; for `CURLY20` and `SCOSINE`, `iARC` reached the time limit; for `INDEF`, `iARC` terminated due to a subproblem solver error; and for all of these four problems, `iR_Newton` terminated due to our minimum step norm limit. The remaining set consisted of 152 test problems with number of variables ranging from 2 to 100,000. For additional details on the problems used and their sizes, see Appendix B.3.

To compare the performance of the implemented algorithms, we generated performance profiles for the number of iterations and number of Hessian-vector products required before

termination. These are shown in Figure 3.1. A performance profile graph of an algorithm at point α shows the fraction of the test set for which the algorithm is able to solve within a factor of 2^α of the best algorithm for the given measure; see [32]. When generating the profiles, we did not include three of the test problems—`CURLY10`, `CURLY30`, and `MODBEALE`—on which `iARC` was unsuccessful while `iR_Newton` was successful. (In particular, `iARC` reached the time limit for all problems.) We feel that this gives a fairer comparison with respect to the problems on which both algorithms were successful.

As seen in Figure 3.1, the algorithms performed relatively comparably when it came to the number of iterations required, though clearly `iR_Newton` had an edge in terms of requiring fewer iterations on various problems. The difference in terms of numbers of Hessian-vector products required was more drastic, and indeed we point to this as the main measure of improved performance for `iR_Newton` versus `iARC`. One reason for this discrepancy is that `iR_Newton` required fewer iterations on some problems. However, more significantly, the difference was due in part to `iR_Newton`'s ability to employ and accept inexact Newton steps (with $\lambda_k = 0$) on many iterations. This is due to the fact that, in CG, one is able to compute the Hessian-vector product $H_k s_k$, needed to check the termination conditions for the computation of s_k, by taking a linear combination of Hessian-vector products already computed in CG; i.e., if $\{p_{k,i}\}$ are the search directions computed in CG such that $s_k = \sum_i \alpha_{k,i} p_{k,i}$, then CG involves computing $H_k p_{k,i}$ for each i and can compute $H_k s_k = \sum_i \alpha_{k,i}(H_k p_{k,i})$. By contrast, one is unable to retrieve this product via a linear combination when the step is computed from the minimization of a cubic function, as is needed in `iARC` and in `iR_Newton` whenever $\sigma_k^{\mathrm{L}} > 0$. Overall, we claim that the primary strength of `iR_Newton` as compared to `iARC` is its ability to employ inexact Newton steps.

For further details of our numerical results, see Appendix B.3. In these results, we also indicate the number of tridiagonal factorizations required; at least one is needed involving a tridiagonal matrix of size $m \times m$ every time an algorithm solves a cubic subproblem over an m-dimensional subspace.

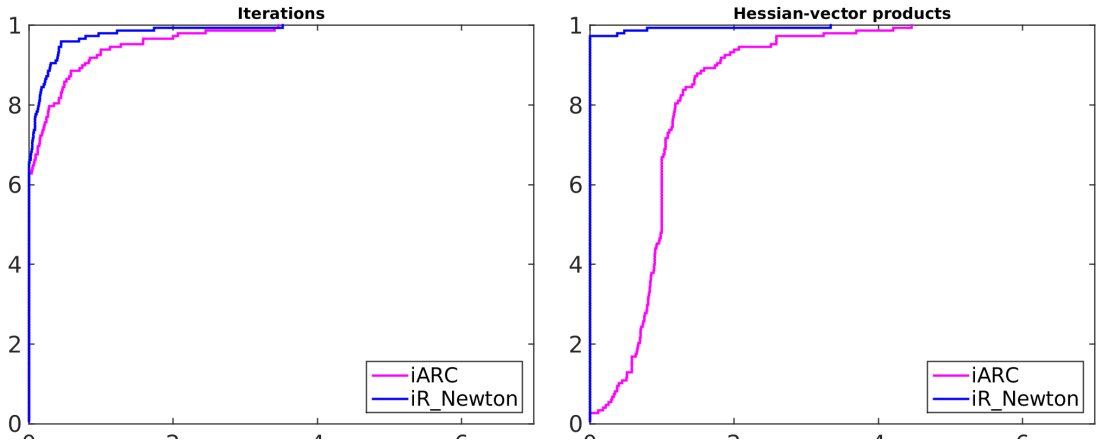


Figure 3.1: Performance profiles for `iARC` and `iR_Newton`.

Chapter 4

Complexity Analysis of a Trust Funnel Algorithm for Equality Constrained Optimization

The purpose of this chapter[1] is to propose a new method for solving equality constrained nonlinear optimization problems. As is well known, such problems are important throughout science and engineering, arising in areas such as network flow optimization [48, 58], optimal allocation with resource constraints [24, 49], maximum likelihood estimations with constraints [47], and optimization with constraints defined by partial differential equations [4, 9, 59].

The algorithm proposed in this chapter can be considered a next step in the design of *practical* algorithms for equality constrained optimization with good worst-case iteration complexity properties. Ours is also a two-phase approach, but is closer to the SQP-type methods representing the state-of-the-art for solving equality constrained problems. In particular, the first phase of our proposed approach follows a trust funnel methodology that locates an ϵ-feasible point in $\mathcal{O}(\epsilon^{-3/2})$ iterations *while also attempting to yield improvements in the objective function.* Borrowing ideas from the trust region method known as TRACE (see Chapter 2), we prove that our method attains the same worst-case iteration complexity bounds as those offered by [6, 19].

[1] A paper containing the original material of this chapter was published in 2018. Please refer to [29].

Organization In the rest of this section, we introduce notation used throughout this chapter and cover preliminary material on equality constrained nonlinear optimization. In §4.2, we motivate and describe our proposed "phase 1" method for locating an ϵ-feasible point while also attempting to reduce the objective function. An analysis of the convergence and worst-case iteration complexity of this phase 1 method is presented in §4.3. Strategies and convergence/complexity guarantees for "phase 2" are given in §4.4. Numerical results are given in §4.5.

Notation A vector with all elements equal to 1 is denoted as e and an identity matrix is denoted as I, where, in each case, the size of the quantity is determined by the context in which it appears. With real symmetric matrices A and B, let $A \succ (\succeq)\ B$ indicate that $A - B$ is positive definite (semidefinite); e.g., $A \succ (\succeq)\ 0$ indicates that A is positive definite (semidefinite). Given vectors $\{u, v\} \subset \mathbb{R}^N$, let $u \perp v$ mean that $u_i v_i = 0$ for all $i \in \{1, 2, \ldots, N\}$. Let $\|x\|$ denote the ℓ_2-norm of x.

4.1 Preliminaries

Given an objective function $f : \mathbb{R}^N \to \mathbb{R}$ and constraint function $c : \mathbb{R}^N \to \mathbb{R}^M$, we study the equality constrained optimization problem

$$\min_{x \in \mathbb{R}^N} \ f(x) \quad \text{s.t.} \quad c(x) = 0. \tag{4.1}$$

At the outset, let us state the following assumption about the problem functions.

Assumption 12. *The functions f and c are twice continuously differentiable.*

In light of Assumption 12, we define $g : \mathbb{R}^N \to \mathbb{R}^N$ as the gradient function of f, i.e., $g := \nabla f$, and define $J : \mathbb{R}^N \to \mathbb{R}^{M \times N}$ as the Jacobian function of c, i.e., $J := \nabla c^T$. The function $c_i : \mathbb{R}^N \to \mathbb{R}$ denotes the ith element of the function c.

Our proposed algorithm follows a local search strategy that merely aims to compute a first-order stationary point for problem (4.1). Defining the Lagrangian $\mathcal{L} : \mathbb{R}^N \times \mathbb{R}^M \to \mathbb{R}$ as given by $\mathcal{L}(x, y) = f(x) + y^T c(x)$, a first-order stationary point (x, y) is one that satisfies $0 = \nabla_x L(x, y) \equiv g(x) + J(x)^T y$ and $0 = \nabla_y L(x, y) \equiv c(x)$.

Our proposed technique for solving problem (4.1) is iterative, generating, amongst other quantities, a sequence of iterates $\{x_k\}$ indexed by $k \in \mathbb{N}$. For ease of exposition, we

also apply an iteration index subscript for function and other quantities corresponding to the kth iteration; e.g., we write f_k to denote $f(x_k)$.

4.2 Phase 1: Obtaining Approximate Feasibility

The goal of phase 1 is to obtain an iterate that is (approximately) feasible. This can, of course, be accomplished by employing an algorithm that focuses exclusively on minimizing a measure of constraint violation. However, we find this idea to be unsatisfactory since such an approach would entirely ignore the objective function. Alternatively, in this section, we present a trust funnel algorithm with good complexity properties for obtaining (approximate) feasibility that attempts to simultaneously reduce the objective f.

4.2.1 Step computation

Similar to other trust funnel algorithms [27, 38], our algorithm employs a step-decomposition approach wherein each trial step is composed of a *normal step* aimed at reducing constraint violation (i.e., infeasibility) and a *tangential step* aimed at reducing the objective function. The algorithm then uses computed information, such as the reductions that the trial step yields in models of the constraint violation and objective function, to determine which of two types of criteria should be used for accepting or rejecting the trial step. To ensure that sufficient priority is given to obtaining (approximate) feasibility, an upper bound on a constraint violation measure is initialized, maintained, and subsequently driven toward zero as improvements toward feasibility are obtained. The algorithm might also nullify the tangential component of a trial step, even after it is computed, if it is deemed too harmful in the algorithm's pursuit toward (approximate) feasibility.

4.2.1.1 Normal step

The purpose of the normal step is to reduce infeasibility. The measure of infeasibility that we employ is $v : \mathbb{R}^N \to \mathbb{R}$ defined by

$$v(x) = \tfrac{1}{2}\|c(x)\|^2. \tag{4.2}$$

At an iterate x_k, the normal step n_k is defined as a minimizer of a second-order Taylor series approximation of v at x_k subject to a trust region constraint, i.e.,

$$n_k \in \arg\min_{n \in \mathbb{R}^N} m_k^v(n) \quad \text{s.t.} \quad \|n\| \leq \delta_k^v, \tag{4.3}$$

where the scalar $\delta_k^v \in (0, \infty)$ is the trust region radius and the model of the constraint violation measure at x_k is $m_k^v : \mathbb{R}^N \to \mathbb{R}$ defined by

$$m_k^v(n) = v_k + g_k^{vT} n + \tfrac{1}{2} n^T H_k^v n \quad \text{with} \quad g_k^v := \nabla v(x_k) = J_k^T c_k, \tag{4.4}$$

$$\text{and} \quad H_k^v := \nabla^2 v(x_k) = J_k^T J_k + \sum_{i=1}^{M} c_i(x_k) \nabla^2 c_i(x_k). \tag{4.5}$$

For any $(x_k, \delta_k^v) \in \mathbb{R}^N \times \mathbb{R}_+$, a globally optimal solution to (4.3) exists [23, Corollary 7.2.2] and n_k has a corresponding dual variable $\lambda_k^v \in \mathbb{R}_+$ such that

$$g_k^v + (H_k^v + \lambda_k^v I) n_k = 0, \tag{4.6a}$$

$$H_k^v + \lambda_k^v I \succeq 0, \tag{4.6b}$$

$$\text{and} \quad 0 \leq \lambda_k^v \perp (\delta_k^v - \|n_k\|) \geq 0. \tag{4.6c}$$

In a standard trust region strategy, a trust region radius is given at the beginning of an iteration, which *explicitly* affects the solution of the subproblem. Indeed, for ease of exposition and analysis, our method is stated in this manner. However, for the purposes of implementation, one might recognize that our method could, in some circumstances—specifically, after any time the normal step trust region radius is contracted—compute a normal step as a solution of (4.3) where the radius is defined *implicitly* by a given dual variable λ_k^v. In particular, given $\lambda_k^v \in [0, \infty)$ that is strictly larger than the negative of the leftmost eigenvalue of H_k^v, one could compute n_k from

$$\mathcal{Q}_k^v(\lambda_k^v) : \quad \min_{n \in \mathbb{R}^N} \; v_k + g_k^{vT} n + \tfrac{1}{2} n^T (H_k^v + \lambda_k^v I) n. \tag{4.7}$$

The unique solution to (4.7), call it $n_k(\lambda_k^v)$, is the solution of the nonsingular linear system $(H_k^v + \lambda_k^v I) n = -g_k^v$, and is the global solution of (4.3) for $\delta_k^v = \|n_k(\lambda_k^v)\|$.

4.2.1.2 Tangential step

The purpose of the tangential step is to reduce the objective function. Specifically, when requested by the algorithm, the tangential step t_k is defined as a minimizer of a quadratic model of the objective function in the null space of the constraint Jacobian subject to a trust region constraint, i.e.,

$$t_k \in \arg\min_{t\in\mathbb{R}^N} m_k^f(n_k + t) \quad \text{s.t.} \quad J_k t = 0 \quad \text{and} \quad \|n_k + t\| \leq \delta_k^s, \tag{4.8}$$

where $\delta_k^s \in (0, \infty)$ is a trust region radius and, with some symmetric $H_k \in \mathbb{R}^{N\times N}$, the objective function model $m_k^f : \mathbb{R}^N \to \mathbb{R}$ is defined by

$$m_k^f(s) = f_k + g_k^T s + \tfrac{1}{2} s^T H_k s. \tag{4.9}$$

Following other trust funnel strategies, one desires δ_k^s to be set such that the trust region describes the area in which the models of the constraint violation and objective function are accurate. In particular, with a trust region radius $\delta_k^f \in (0, \infty)$ for the objective function, our algorithm employs, for some scalar $\kappa_\delta \in (1, \infty)$, the value

$$\delta_k^s := \min\{\kappa_\delta \delta_k^v, \delta_k^f\}. \tag{4.10}$$

Due to this choice of trust region radius, it is deemed not worthwhile to compute a nonzero tangential step if the feasible region of (4.8) is small. Specifically, our algorithm only computes a nonzero t_k when $\|n_k\| \leq \kappa_n \delta_k^s$ for some $\kappa_n \in (0, 1)$. In addition, it only makes sense to compute a tangential step when reasonable progress in reducing f in the null space of J_k can be expected. To predict the potential progress, we define

$$g_k^p := Z_k Z_k^T (g_k + H_k n_k), \tag{4.11}$$

where the columns of Z_k form an orthonormal basis for $\text{Null}(J_k)$. If $\|g_k^p\| < \kappa_p \|g_k^v\|$ for some $\kappa_p \in (0, \infty)$, then computing a tangential step is not worthwhile and we simply set the primal-dual solution (estimate) for (4.8) to zero.

For any $(x_k, \delta_k^s, H_k) \in \mathbb{R}^N \times \mathbb{R}_+ \times \mathbb{R}^{N\times N}$, a globally optimal solution to (4.8) exists [23, Corollary 7.2.2] and t_k has corresponding dual variables $y_k^f \in \mathbb{R}^M$ and $\lambda_k^f \in \mathbb{R}_+$ (for the

null space and trust region constraints, respectively) such that

$$\begin{bmatrix} H_k + \lambda_k^f I & J_k^T \\ J_k & 0 \end{bmatrix} \begin{bmatrix} t_k \\ y_k^f \end{bmatrix} = - \begin{bmatrix} g_k + (H_k + \lambda_k^f I) n_k \\ 0 \end{bmatrix}, \tag{4.12a}$$

$$Z_k^T H_k Z_k + \lambda_k^f I \succeq 0, \tag{4.12b}$$

$$\text{and} \quad 0 \leq \lambda_k^f \perp (\delta_k^s - \|n_k + t_k\|) \geq 0. \tag{4.12c}$$

Similarly as for the normal step computation, an implementation of our algorithm could compute t_k not as a solution of (4.8) for a given δ_k^s, but as a solution of a regularized subproblem for a given dual variable for the trust region constraint. Specifically, for $\lambda_k^f \in [0, \infty)$ that is strictly larger than the negative of the leftmost eigenvalue of $Z_k^T H_k Z_k$, one could solve the following subproblem for the tangential step:

$$\mathcal{Q}_k^f(\lambda_k^f): \quad \min_{t \in \mathbb{R}^N} \ (g_k + (H_k + \lambda_k^f I) n_k)^T t + \tfrac{1}{2} t^T (H_k + \lambda_k^f I) t \quad \text{s.t.} \quad J_k t = 0. \tag{4.13}$$

The unique solution $t_k(\lambda_k^f)$ of (4.13) is a global solution of (4.8) for $\delta_k^s = \|n_k + t_k(\lambda_k^f)\|$.

There are situations in which our algorithm discards a computed tangential step after one is computed, i.e., situations when the algorithm resets $t_k \leftarrow 0$. Specifically, this occurs when any of the following conditions fails to hold:

$$m_k^v(0) - m_k^v(n_k + t_k) \geq \kappa_{vm} \left(m_k^v(0) - m_k^v(n_k) \right) \qquad \text{for some} \quad \kappa_{vm} \in (0, 1); \tag{4.14a}$$

$$\|n_k + t_k\| \geq \kappa_{ntn} \|n_k\| \qquad \text{for some} \quad \kappa_{ntn} \in (0, 1); \tag{4.14b}$$

$$\|H_k^v t_k\| \leq \kappa_{ht} \|n_k + t_k\|^2 \qquad \text{for some} \quad \kappa_{ht} \in (0, \infty). \tag{4.14c}$$

The first of these conditions requires that the reduction in the constraint violation model for the full step $s_k := n_k + t_k$ is sufficiently large with respect to that obtained by the normal step; the second requires that the full step is sufficiently large in norm compared to the normal step; and the third requires that the action of the tangential step on the Hessian of the constraint violation model is not too large compared to the squared norm of the full step. It is worthwhile to mention that all of these conditions are satisfied automatically when $H_k^v = J_k^T J_k$ (recall (4.5)), which occurs, e.g., when c is affine. However, since this does not hold in general, our algorithm requires these conditions explicitly, or else resets the tangential step to zero (which satisfies (4.14)).

4.2.2 Step acceptance

After computing a normal step n_k and potentially a tangential step t_k, the algorithm determines whether to accept the full step $s_k := n_k + t_k$. The strategy that it employs is based on first using the obtained reductions in the models of constraint violation and the objective, as well as other related quantities, to determine what should be the main goal of the iteration: reducing constraint violation or the objective function. Since the primary goal of phase 1 is to obtain (approximate) feasibility, the algorithm has a preference toward reducing constraint violation unless the potential reduction in the objective function is particularly compelling. Specifically, if the following conditions hold, indicating good potential progress in reducing the objective, then the algorithm performs an F-ITERATION (see §4.2.2.1):

$$
\begin{aligned}
t_k \neq 0 \quad \text{with} \quad \|t_k\| &\geq \kappa_{st}\|s_k\| && \text{for some } \kappa_{st} \in (0,1), && \text{(4.15a)}\\
m_k^f(0) - m_k^f(s_k) &\geq \kappa_{fm}(m_k^f(n_k) - m_k^f(s_k)), && \text{for some } \kappa_{fm} \in (0,1), && \text{(4.15b)}\\
v(x_k + s_k) &\leq v_k^{\max} - \kappa_\rho\|s_k\|^3 && \text{for some } \kappa_\rho \in (0,1), && \text{(4.15c)}\\
n_k^T t_k &\geq -\tfrac{1}{2}\kappa_{ntt}\|t_k\|^2 && \text{for some } \kappa_{ntt} \in (0,1), && \text{(4.15d)}\\
\lambda_k^v &\leq \sigma_k^v\|n_k\| && \text{and} && \text{(4.15e)}\\
\left\|(H_k - \nabla^2 f(x_k))s_k\right\| &\leq \kappa_{hs}\|s_k\|^2 && \text{for some } \kappa_{hs} \in (0,\infty). && \text{(4.15f)}
\end{aligned}
$$

Conditions (4.15a)–(4.15c) are similar to those employed in other trust funnel algorithms, except that (4.15a) and (4.15c) are stronger (than the common, weaker requirements that $t_k \neq 0$ and $v(x_k + s_k) \leq v_k^{\max}$). Employed here is a scalar sequence $\{v_k^{\max}\}$ updated dynamically by the algorithm that represents an upper bound on constraint violation; for this sequence, the algorithm ensures (see Lemma 42) that $v_k \leq v_k^{\max}$ and $v_{k+1}^{\max} \leq v_k^{\max}$ for all $k \in \mathbb{N}$. Condition (4.15d) ensures that, for an F-ITERATION, the inner product between the normal and tangential steps is not too negative (or else the tangential step might undo too much of the progress toward feasibility offered by the normal step). Finally, conditions (4.15e) and (4.15f) are essential for achieving good complexity properties, requiring that any F-ITERATION involves a normal step that is sufficiently large compared to the Lagrange multiplier for the trust region constraint and that the action of the full step on H_k does not differ too much from its action on $\nabla^2 f(x_k)$. If any condition in (4.15) does not hold, then a V-ITERATION is perfomed (see §4.2.2.2).

Viewing (4.15f), it is worthwhile to reflect on the choice of H_k in the algorithm. With the attainment of optimality (not only feasibility) in mind, standard practice would suggest that it is desirable to choose H_k as the Hessian of the Lagrangian of problem (4.1) for some multiplier vector $y_k \in \mathbb{R}^M$. This multiplier vector could be obtained, e.g., as the *QP multipliers* from some previous iteration or *least squares multipliers* using current derivative information. For our purposes of obtaining good complexity properties for phase 1, we do not require a particular choice of H_k, but this discussion and (4.15f) do offer some guidance. Specifically, one might choose H_k as an approximation of the Hessian of the Lagrangian, potentially with the magnitude of the multiplier vector restricted in such a way that, after the full step is computed, the action of it on H_k will not differ too much with its action on $\nabla^2 f(x_k)$. This is more reasonable to do when it is known that the set of optimal multipliers is bounded (which is true, e.g., under various constraint qualifications). In any case, setting H_k to $\nabla f^2(x_k)$ is at least one valid choice as far as our analysis is concerned.

4.2.2.1 F-iteration

If (4.15) holds, then we determine that the kth iteration is an F-ITERATION. In this case, we begin by calculating the quantity

$$\rho_k^f \leftarrow (f_k - f(x_k + s_k))/\|s_k\|^3, \tag{4.16}$$

which measures decrease in f. Using this quantity, acceptance or rejection of the step and the rules for updating the trust region radius are similar as in Algorithm 6. As for the trust funnel radius, rather than the update in [38, Algorithm 2.1], we require a modified update; in particular, we use, for some $\{\kappa_{v1}, \kappa_{v2}\} \subset (0, 1)$,

$$v_{k+1}^{\max} \leftarrow \min\{\max\{\kappa_{v1} v_k^{\max}, v_k^{\max} - \kappa_\rho \|s_k\|^3\}, v_{k+1} + \kappa_{v2}(v_k^{\max} - v_{k+1})\}. \tag{4.17}$$

4.2.2.2 V-iteration

When any one of the conditions in (4.15) does not hold, the kth iteration is a V-ITERATION, during which the main focus is to decrease the measure of infeasibility v. In this case, we calculate

$$\rho_k^v \leftarrow (v(x_k) - v(x_k + s_k))/\|s_k\|^3, \tag{4.18}$$

which provides a measure of the decrease in constraint violation. The rules for accepting or rejecting the trial step and for updating the trust region radius are the same as those in Algorithm 6. One addition is that during a successful V-ITERATION, the trust funnel radius is updated, using the same constants as in (4.17), as

$$v_{k+1}^{\max} \leftarrow \min\{\max\{\kappa_{v1} v_k^{\max}, v_{k+1} + \kappa_{v2}(v_k - v_{k+1})\}, v_{k+1} + \kappa_{v2}(v_k^{\max} - v_{k+1})\}. \tag{4.19}$$

4.2.3 Algorithm statement

Our complete algorithm for finding an (approximately) feasible point can now be stated as Algorithm 8 on page 104, which in turn calls the F-ITERATION subroutine stated as Algorithm 9 on page 105 and the V-ITERATION subroutine stated as Algorithm 10 on page 106.

4.3 Convergence and Complexity Analyses for Phase 1

The analyses that we present require the following assumption related to the iterate sequence.

Assumption 13. *The sequence of iterates $\{x_k\}$ is contained in a compact set. In addition, the sequence $\{\|H_k\|\}$ is bounded over $k \in \mathbb{N}$.*

Our analysis makes extensive use of the following mutually exclusive and exhaustive subsets of the iteration index sequence generated by Algorithm 8:

$$\begin{aligned} \mathcal{I} &:= \{k \in \mathbb{N} : \|g_k^v\| > \epsilon\}, \\ \mathcal{F} &:= \{k \in \mathcal{I} : \text{iteration } k \text{ is an F-ITERATION}\}, \\ \text{and } \mathcal{V} &:= \{k \in \mathcal{I} : \text{iteration } k \text{ is a V-ITERATION}\}. \end{aligned}$$

It will also be convenient to define the index set of iterations for which tangential steps are computed and not reset to zero by our method:

$$\begin{aligned} \mathcal{I}^t &:= \{k \in \mathcal{I} : t_k \neq 0 \text{ when Step 7 of Algorithm 8 is reached}\} \\ &= \{k \in \mathcal{I} : \text{ Step 22 of Algorithm 8 is reached and all conditions in (4.14) hold}\}. \end{aligned}$$

Algorithm 8 Trust Funnel Algorithm for Phase 1

Require: $\{\kappa_n, \kappa_{vm}, \kappa_{ntn}, \kappa_\rho, \kappa_{fm}, \kappa_{st}, \kappa_{ntt}, \kappa_{v1}, \kappa_{v2}, \gamma_C\} \subset (0,1)$,
$\{\kappa_p, \kappa_{ht}, \kappa_{hs}, \epsilon, \underline{\sigma}\} \subset (0,\infty)$, $\{\kappa_\delta, \gamma_E, \gamma_\lambda\} \in (1,\infty)$, and $\overline{\sigma} \in [\underline{\sigma}, \infty)$;
F-ITERATION (Algorithm 9, page 105) and V-ITERATION (Algorithm 10, page 106)

procedure TRUST_FUNNEL
  choose $x_0 \in \mathbb{R}^N$, $v_0^{\max} \in [\max\{1, v_0\}, \infty)$, and $\sigma_0^v \in [\underline{\sigma}, \overline{\sigma}]$
  choose $\{\delta_0^v, \Delta_0^v, \delta_0^f\} \subset (0,\infty)$ such that $\delta_0^v \le \Delta_0^v$
  for $k \in \mathbb{N}$ **do**
    if $\|g_k^v\| \le \epsilon$ **then return** x_k
    $(n_k, t_k, \lambda_k^v, \lambda_k^f) \leftarrow$ COMPUTE_STEPS$(x_k, \delta_k^v, \delta_k^s)$
    set $s_k \leftarrow n_k + t_k$
    set $\sigma_k^v \leftarrow$ COMPUTE_SIGMA$(n_k, \lambda_k^v, \sigma_{k-1}^v, \rho_{k-1}^v)$
    if (4.15) is satisfied **then**
      set ρ_k^f by (4.16)
      $(x_{k+1}, v_{k+1}^{\max}, \delta_{k+1}^f) \leftarrow$ F-ITERATION$(x_k, n_k, s_k, v_k^{\max}, \delta_k^f, \lambda_k^f, \rho_k^f)$
      set $\delta_{k+1}^v \leftarrow \delta_k^v$, $\Delta_{k+1}^v \leftarrow \Delta_k^v$, and $\rho_k^v \leftarrow \infty$
    else
      set ρ_k^v by (4.18)
      $(x_{k+1}, v_{k+1}^{\max}, \delta_{k+1}^v, \Delta_{k+1}^v) \leftarrow$ V-ITERATION$(x_k, n_k, s_k, v_k^{\max}, \delta_k^v, \Delta_k^v, \lambda_k^v, \sigma_k^v, \rho_k^v)$
      set $\delta_{k+1}^f \leftarrow \delta_k^f$ and $\rho_k^f \leftarrow \infty$

procedure COMPUTE_STEPS$(x_k, \delta_k^v, \delta_k^s)$
  set (n_k, λ_k^v) as a primal-dual solution to (4.3)
  set $(t_k, \lambda_k^f) \leftarrow (0,0)$
  if $\|n_k\| \le \kappa_n \delta_k^s$ and $\|g_k^p\| \ge \kappa_p \|g_k^v\|$ **then**
    set $(t_k, y_k^f, \lambda_k^f)$ as a primal-dual solution to (4.8)
    if any condition in (4.14) fails to hold **then** set $(t_k, \lambda_k^f) \leftarrow (0,0)$
  return $(n_k, t_k, \lambda_k^v, \lambda_k^f)$

procedure COMPUTE_SIGMA$(n_k, \lambda_k^v, \sigma_{k-1}^v, \rho_{k-1}^v)$
  if iteration $(k-1)$ was an F-ITERATION **then**
    set $\sigma_k^v \leftarrow \sigma_{k-1}^v$
  else
    if $\rho_{k-1}^v < \kappa_\rho$ **then** set $\sigma_k^v \leftarrow \max\{\sigma_{k-1}^v, \lambda_k^v / \|n_k\|\}$ **else** set $\sigma_k^v \leftarrow \sigma_{k-1}^v$
  return σ_k^v

4.3.1 Convergence analysis for phase 1

The goal of our convergence analysis is to prove that Algorithm 8 terminates finitely, i.e., $|\mathcal{I}| < \infty$. Our analysis to prove this fact requires a refined examination of the subsets $\mathcal{F}$ and $\mathcal{V}$ of $\mathcal{I}$. For these purposes, we define disjoint subsets of $\mathcal{F}$ as $\mathcal{S}^f := \{k \in \mathcal{F} : \rho_k^f \ge \kappa_\rho\}$

Algorithm 9 F-ITERATION subroutine

procedure F-ITERATION$(x_k, n_k, s_k, v_k^{\max}, \delta_k^f, \lambda_k^f, \rho_k^f)$
  if $\rho_k^f \geq \kappa_\rho$ **then** [accept step]
    set $x_{k+1} \leftarrow x_k + s_k$
    set $v_{k+1}^{\max}$ according to (4.17)
    set $\delta_{k+1}^f \leftarrow \max\{\delta_k^f, \gamma_{\text{E}}\|s_k\|\}$
  else (i.e., **if** $\rho_k^f < \kappa_\rho$) [contract trust region]
    set $x_{k+1} \leftarrow x_k$
    set $v_{k+1}^{\max} \leftarrow v_k^{\max}$
    set $\delta_{k+1}^f \leftarrow$ F-CONTRACT$(n_k, s_k, \delta_k^f, \lambda_k^f)$
  return $(x_{k+1}, v_{k+1}^{\max}, \delta_{k+1}^f)$

procedure F-CONTRACT$(n_k, s_k, \delta_k^f, \lambda_k^f)$
  if $\lambda_k^f < \underline{\sigma}\|s_k\|$ **then**
    set $\lambda^f > \lambda_k^f$ so the solution $t(\lambda^f)$ of $\mathcal{Q}_k^f(\lambda^f)$ yields $\underline{\sigma} \leq \lambda^f / \|n_k + t(\lambda^f)\|$
    return $\delta_{k+1}^f \leftarrow \|n_k + t(\lambda^f)\|$
  else (i.e., **if** $\lambda_k^f \geq \underline{\sigma}\|s_k\|$)
    return $\delta_{k+1}^f \leftarrow \gamma_{\text{C}}\|s_k\|$

and $\mathcal{C}^f := \{k \in \mathcal{F} : \rho_k^f < \kappa_\rho\}$, and disjoint subsets of $\mathcal{V}$ as

$$\mathcal{S}^v := \{k \in \mathcal{V} : \rho_k^v \geq \kappa_\rho \text{ and either } \lambda_k^v \leq \sigma_k^v\|n_k\| \text{ or } \|n_k\| = \Delta_k^v\},$$
$$\mathcal{C}^v := \{k \in \mathcal{V} : \rho_k^v < \kappa_\rho\} \quad \text{and} \quad \mathcal{E}^v := \{k \in \mathcal{V} : k \notin \mathcal{S}^v \cup \mathcal{C}^v\}.$$

We further partition the set $\mathcal{S}^v$ into the subsets $\mathcal{S}_\Delta^v := \{k \in \mathcal{S}^v : \|n_k\| = \Delta_k^v\}$ and $\mathcal{S}_\sigma^v := \{k \in \mathcal{S}^v : k \notin \mathcal{S}_\Delta^v\}$. Finally, for convenience, we also define the unions $\mathcal{S} := \{k \in \mathcal{I} : k \in \mathcal{S}^f \cup \mathcal{S}^v\}$ and $\mathcal{C} := \{k \in \mathcal{I} : k \in \mathcal{C}^f \cup \mathcal{C}^v\}$. Due to the updates for the primal iterate and/or trust region radii, we refer to iterations with indices in $\mathcal{S}$ as successful, with indices in $\mathcal{C}$ as contractions, and with indices in $\mathcal{E}^v$ as expansions.

Basic relationships between all of these sets are summarized in our first lemma.

Lemma 39. *The following relationships hold:*

(i) $\mathcal{F} \cap \mathcal{V} = \emptyset$ *and* $\mathcal{F} \cup \mathcal{V} = \mathcal{I}$;

(ii) $\mathcal{S}^f \cap \mathcal{C}^f = \emptyset$ *and* $\mathcal{S}^f \cup \mathcal{C}^f = \mathcal{F}$;

(iii) $\mathcal{S}^v$, $\mathcal{C}^v$, *and* $\mathcal{E}^v$ *are mutually disjoint and* $\mathcal{S}^v \cup \mathcal{C}^v \cup \mathcal{E}^v = \mathcal{V}$; *and*

(iv) if $k \in \mathcal{I} \setminus \mathcal{I}^t$, *then* $k \in \mathcal{V}$.

Algorithm 10 V-ITERATION subroutine

procedure V-ITERATION$(x_k, n_k, s_k, v_k^{\max}, \delta_k^v, \Delta_k^v, \lambda_k^v, \sigma_k^v, \rho_k^v)$
  if $\rho_k^v \geq \kappa_\rho$ and either $\lambda_k^v \leq \sigma_k^v \|n_k\|$ or $\|n_k\| = \Delta_k^v$ **then** [accept step]
    set $x_{k+1} \leftarrow x_k + s_k$
    set $v_{k+1}^{\max}$ according to (4.19)
    set $\Delta_{k+1}^v \leftarrow \max\{\Delta_k^v, \gamma_{\text{E}} \|n_k\|\}$
    set $\delta_{k+1}^v \leftarrow \min\{\Delta_{k+1}^v, \max\{\delta_k^v, \gamma_{\text{E}} \|n_k\|\}\}$
  else if $\rho_k^v < \kappa_\rho$ **then** [contract trust region]
    set $x_{k+1} \leftarrow x_k$
    set $v_{k+1}^{\max} \leftarrow v_k^{\max}$
    set $\Delta_{k+1}^v \leftarrow \Delta_k^v$
    set $\delta_{k+1}^v \leftarrow$ V-CONTRACT$(n_k, s_k, \delta_k^v, \lambda_k^v)$
  else (i.e., **if** $\rho_k^v \geq \kappa_\rho$, $\lambda_k^v > \sigma_k^v \|n_k\|$, and $\|n_k\| < \Delta_k^v$) [expand trust region]
    set $x_{k+1} \leftarrow x_k$
    set $v_{k+1}^{\max} \leftarrow v_k^{\max}$
    set $\Delta_{k+1}^v \leftarrow \Delta_k^v$
    set $\delta_{k+1}^v \leftarrow \min\{\Delta_{k+1}^v, \lambda_k^v / \sigma_k^v\}$
  return $(x_{k+1}, v_{k+1}^{\max}, \delta_{k+1}^v, \Delta_{k+1}^v)$

procedure V-CONTRACT$(n_k, s_k, \delta_k^v, \lambda_k^v)$
  if $\lambda_k^v < \underline{\sigma} \|n_k\|$ **then**
    set $\hat{\lambda}^v \leftarrow \lambda_k^v + (\underline{\sigma} \|g_k^v\|)^{1/2}$
    set $\lambda^v \leftarrow \hat{\lambda}^v$
    set $n(\lambda^v)$ as the solution of $\mathcal{Q}_k^v(\lambda^v)$
    if $\lambda^v / \|n(\lambda^v)\| \leq \overline{\sigma}$ **then**
      return $\delta_{k+1}^v \leftarrow \|n(\lambda^v)\|$
    else
      set $\lambda^v \in (\lambda_k^v, \hat{\lambda}^v)$ so the solution $n(\lambda^v)$ of $\mathcal{Q}_k^v(\lambda^v)$ yields $\underline{\sigma} \leq \lambda^v / \|n(\lambda^v)\| \leq \overline{\sigma}$
      return $\delta_{k+1}^v \leftarrow \|n(\lambda^v)\|$
  else (i.e., **if** $\lambda_k^v \geq \underline{\sigma} \|n_k\|$)
    set $\lambda^v \leftarrow \gamma_\lambda \lambda_k^v$
    set $n(\lambda^v)$ as the solution of $\mathcal{Q}_k^v(\lambda^v)$
    if $\|n(\lambda^v)\| \geq \gamma_{\text{C}} \|n_k\|$ **then**
      return $\delta_{k+1}^v \leftarrow \|n(\lambda^v)\|$
    else
      return $\delta_{k+1}^v \leftarrow \gamma_{\text{C}} \|n_k\|$

Proof. The fact that $\mathcal{F} \cap \mathcal{V} = \emptyset$ follows from the two cases resulting from the conditional statement in Step 9 of Algorithm 8. The rest of part (i), part (ii), and part (iii) follow from the definitions of the relevant sets. Part (iv) can be seen to hold as follows. If $k \in \mathcal{I} \setminus \mathcal{I}^t$, then $t_k = 0$ so that (4.15a) does not hold. It now follows from the logic in Algorithm 8 that $k \in \mathcal{V}$ as claimed. □

The results in the next lemma are consequences of Assumptions 12 and 13.

Lemma 40. *The following hold:*

(i) there exists $\theta_{fc} \in (1, \infty)$ *so* $\max\{\|g_k\|, \|c_k\|, \|J_k\|, \|H_k^v\|\} \leq \theta_{fc}$ *for all* $k \in \mathcal{I}$*;*

(ii) $\|g_k^v\| \equiv \|J_k^T c_k\| \leq \theta_{fc}\|c_k\|$ *for all* $k \in \mathcal{I}$*; and*

(iii) $g^v : \mathbb{R}^N \to \mathbb{R}^N$ *defined by* $g^v(x) = J(x)^T c(x)$ *(recall (4.4)) is Lipschitz continuous with Lipschitz constant* $g_{Lip}^v > 0$ *over an open set containing* $\{x_k\}$*.*

Proof. Part (i) follows from Assumptions 12 and 13. Part (ii) follows since, by the Cauchy–Schwarz inequality, $\|J_k^T c_k\| \leq \|J_k\|\|c_k\| \leq \theta_{fc}\|c_k\|$. Part (iii) follows since the first derivative of g^v is uniformly bounded under Assumptions 12 and 13. □

We now summarize properties associated with the normal and tangential steps.

Lemma 41. *The following hold for all* $k \in \mathcal{I}$*:*

(i) $n_k \neq 0$ *and* $s_k \neq 0$*; and*

(ii) in Step 7 of Algorithm 8, the vector t_k *satisfies (4.14).*

Proof. We first prove part (i). Since $k \in \mathcal{I}$, it follows that $\|g_k^v\| > \epsilon$, which combined with (4.6a) implies that $n_k \neq 0$, as claimed. Now, in order to derive a contradiction, suppose that $0 = s_k = n_k + t_k$, which means that $-t_k = n_k \neq 0$. From $g_k^v \neq 0$ and (4.6a), it follows that $(H_k^v + \lambda_k^v I)n_k = -g_k^v \neq 0$, which gives

$$n_k^T(H_k^v + \lambda_k^v I)n_k = -n_k^T g_k^v = -n_k^T J_k^T c_k = -(J_k n_k)^T c_k = 0, \tag{4.20}$$

where the last equality follows from $n_k = -t_k$ and $J_k t_k = 0$ (see (4.12a)). It now follows from (4.20), symmetry of $H_k^v + \lambda_k^v I$, and (4.6b) that $0 = (H_k^v + \lambda_k^v I)n_k = -g_k^v$, which is a contradiction. This completes the proof of part (i).

To prove part (ii), first observe that the conditions in (4.14) are trivially satisfied if $t_k = 0$. On the other hand, if Step 7 is reached with $t_k \neq 0$, then Step 22 must have been reached, at which point it must have been determined that all of the conditions in (4.14) held true (or else t_k would have been reset to the zero vector). □

Next, we show that $\{v_k^{\max}\}$ is a monotonically decreasing bound for $\{v_k\}$.

Lemma 42. *For all $k \in \mathcal{I}$, it follows that $v_k \leq v_k^{\max}$ and $0 < v_{k+1}^{\max} \leq v_k^{\max}$.*

Proof. The result holds trivially if $\mathcal{I} = \emptyset$. Thus, let us assume that $\mathcal{I} \neq \emptyset$, which ensures that $0 \in \mathcal{I}$. Let us now use induction to prove the first inequality, as well as positivity of $v_k^{\max}$ for all $k \in \mathcal{I}$. From the initialization in Algorithm 8, it follows that $v_0 \leq v_0^{\max}$ and $v_0^{\max} > 0$. Now, to complete the induction step, let us assume that $v_k \leq v_k^{\max}$ and $v_k^{\max} > 0$ for some $k \in \mathcal{I}$, then consider three cases.

Case 1: $k \in \mathcal{S}^f$. When $k \in \mathcal{S}^f$, let us consider the two possibilities based on the procedure for setting $v_{k+1}^{\max}$ stated in (4.17). If (4.17) sets $v_{k+1}^{\max} = v_{k+1} + \kappa_{v2}(v_k^{\max} - v_{k+1})$, then the fact that $k \in \mathcal{S}^f \subseteq \mathcal{F}$, (4.15c), and Lemma 41(i) imply that

$$v_{k+1}^{\max} = v_{k+1} + \kappa_{v2}(v_k^{\max} - v_{k+1}) \geq v_{k+1} + \kappa_{v2}\kappa_\rho \|s_k\|^3 > v_{k+1} \geq 0.$$

On the other hand, if (4.17) sets $v_{k+1}^{\max} = \max\{\kappa_{v1} v_k^{\max}, v_k^{\max} - \kappa_\rho \|s_k\|^3\}$, then using the induction hypothesis, the fact that $k \in \mathcal{S}^f \subseteq \mathcal{F}$, and (4.15c), it follows that

$$v_{k+1}^{\max} \geq \kappa_{v1} v_k^{\max} > 0 \quad \text{and} \quad v_{k+1}^{\max} \geq v_k^{\max} - \kappa_\rho \|s_k\|^3 \geq v_{k+1} \geq 0.$$

This case is complete since, in each scenario, $v_{k+1}^{\max} \geq v_{k+1}$ and $v_{k+1}^{\max} > 0$.

Case 2: $k \in \mathcal{S}^v$. When $k \in \mathcal{S}^v$, let us consider the two possibilities based on the procedure for setting $v_{k+1}^{\max}$ stated in (4.19). If (4.19) sets $v_{k+1}^{\max} = v_{k+1} + \kappa_{v2}(v_k^{\max} - v_{k+1})$, then it follows from the induction hypothesis and the fact that $\rho_k^v \geq \kappa_\rho$ for $k \in \mathcal{S}^v$ (which, in particular, implies that $v_{k+1} < v_k$ for $k \in \mathcal{S}^v$) that

$$v_{k+1}^{\max} = v_{k+1} + \kappa_{v2}(v_k^{\max} - v_{k+1}) \geq v_{k+1} + \kappa_{v2}(v_k - v_{k+1}) > v_{k+1} \geq 0.$$

On the other hand, if (4.19) sets $v_{k+1}^{\max} = \max\{\kappa_{v1} v_k^{\max}, v_{k+1} + \kappa_{v2}(v_k - v_{k+1})\}$, then the induction hypothesis and the fact that $v_{k+1} < v_k$ for $k \in \mathcal{S}^v$ implies that

$$v_{k+1}^{\max} \geq \kappa_{v1} v_k^{\max} > 0 \quad \text{and} \quad v_{k+1}^{\max} \geq v_{k+1} + \kappa_{v2}(v_k - v_{k+1}) > v_{k+1} \geq 0.$$

This case is complete since, in each scenario, $v_{k+1}^{\max} \geq v_{k+1}$ and $v_{k+1}^{\max} > 0$.

Case 3: $k \notin \mathcal{S}^f \cup \mathcal{S}^v$. When $k \notin \mathcal{S}^f \cup \mathcal{S}^v$, it follows that $k \in \mathcal{C} \cup \mathcal{E}^v$, which may be combined with the induction hypothesis and the updating procedures for x_k and $v_k^{\max}$ in Algorithms 9 and 10 to deduce that $0 < v_k^{\max} = v_{k+1}^{\max}$ and $v_{k+1} = v_k \leq v_k^{\max} = v_{k+1}^{\max}$.

Combining the conclusions of the three cases above, it follows by induction that the first inequality of the lemma holds true and $v_k^{\max} > 0$ for all $k \in \mathcal{I}$.

Let us now prove that $v_{k+1}^{\max} \leq v_k^{\max}$ for all $k \in \mathcal{I}$, again by considering three cases. First, if $k \in \mathcal{S}^f$, then $v_{k+1}^{\max}$ is set using (4.17) such that

$$v_{k+1}^{\max} \leq \max\{\kappa_{v1} v_k^{\max}, v_k^{\max} - \kappa_\rho \|s_k\|^3\} < v_k^{\max},$$

where the strict inequality follows by $\kappa_{v1} \in (0,1)$ and Lemma 41(i). Second, if $k \in \mathcal{S}^v$, then $v_{k+1} < v_k \leq v_k^{\max}$, where we have used the proved fact that $v_k \leq v_k^{\max}$; thus, $v_k^{\max} - v_{k+1} > 0$. Then, since $v_{k+1}^{\max}$ is set using (4.19), it follows that

$$v_k^{\max} - v_{k+1}^{\max} \geq v_k^{\max} - v_{k+1} - \kappa_{v2}(v_k^{\max} - v_{k+1}) = (1 - \kappa_{v2})(v_k^{\max} - v_{k+1}) > 0.$$

Third, if $k \notin \mathcal{S}^f \cup \mathcal{S}^v$, then, by construction in Algorithms 9 and 10, $v_{k+1}^{\max} = v_k^{\max}$. □

Our next lemma gives a lower bound for the decrease in the trust funnel radius.

Lemma 43. *If $k \in \mathcal{S}$, then $v_k^{\max} - v_{k+1}^{\max} \geq \kappa_\rho (1 - \kappa_{v2}) \|s_k\|^3$.*

Proof. If $k \in \mathcal{S}^f$, then $v_{k+1}^{\max}$ is set using (4.17). In this case,

$$\begin{aligned} v_k^{\max} - v_{k+1}^{\max} &\geq v_k^{\max} - v_{k+1} - \kappa_{v2}(v_k^{\max} - v_{k+1}) \\ &= (1 - \kappa_{v2})(v_k^{\max} - v_{k+1}) \geq \kappa_\rho (1 - \kappa_{v2}) \|s_k\|^3, \end{aligned}$$

where the last inequality follows from (4.15c) (since $k \in \mathcal{S}^f \subseteq \mathcal{F}$). If $k \in \mathcal{S}^v$, then $v_{k+1}^{\max}$ is set using (4.19). In this case, by Lemma 42 and the fact that $\rho_k^v \geq \kappa_\rho$ for $k \in \mathcal{S}^v$,

$$\begin{aligned} v_k^{\max} - v_{k+1}^{\max} &\geq v_k^{\max} - v_{k+1} - \kappa_{v2}(v_k^{\max} - v_{k+1}) \\ &= (1 - \kappa_{v2})(v_k^{\max} - v_{k+1}) \geq (1 - \kappa_{v2})(v_k - v_{k+1}) \geq \kappa_\rho (1 - \kappa_{v2}) \|s_k\|^3, \end{aligned}$$

which completes the proof. □

Subsequently in our analysis, it will be convenient to consider an alternative formulation of problem (4.8) that arises from an orthogonal decomposition of the normal step n_k into its projection onto the range space of J_k^T, call it n_k^R, and its projection onto the null

space of J_k, call it n_k^N. Specifically, considering

$$t_k^N \in \arg\min_{t^N \in \mathbb{R}^N} m_k^f(n_k^R + t^N) \quad \text{s.t.} \quad J_k t^N = 0 \quad \text{and} \quad \|t^N\| \leq \sqrt{(\delta_k^s)^2 - \|n_k^R\|^2}, \tag{4.21}$$

we can recover the solution of (4.8) as $t_k \leftarrow t_k^N - n_k^N$. Similarly, for any $\lambda_k^f \in [0, \infty)$ that is strictly greater than the left-most eigenvalue of $Z_k^T H_k Z_k$, let us define

$$\bar{\mathcal{Q}}_k^f(\lambda_k^f): \quad \min_{t^N \in \mathbb{R}^N} (g_k + H_k n_k^R)^T t^N + \tfrac{1}{2}(t^N)^T (H_k + \lambda_k^f I) t^N \quad \text{s.t.} \quad J_k t^N = 0. \tag{4.22}$$

In the next lemma, we formally establish the equivalence between problems (4.21) and (4.8), as well as between problems (4.22) and (4.13).

Lemma 44. *For all $k \in \mathcal{I}$, the following problem equivalences hold:*

(i) if $\|n_k\| \leq \delta_k^s$, then problems (4.21) and (4.8) are equivalent in that (t_k^N, λ_k^N) is part of a primal-dual solution of problem (4.21) if and only if $(t_k, \lambda_k^f) = (t_k^N - n_k^N, \lambda_k^N)$ is part of a primal-dual solution of problem (4.8); and

(ii) if $Z_k^T H_k Z_k + \lambda_k^f I \succ 0$, then problems (4.22) and (4.13) are equivalent in that t_k^N solves problem (4.22) if and only if $t_k = t_k^N - n_k^N$ solves problem (4.13).

Proof. To prove part (i), first note that $\|n_k\| \leq \delta_k^s$ ensures that problems (4.21) and (4.8) are feasible. Then, by $J_k t^N = 0$ in (4.21), the vector $n_k^R \in \text{Range}(J_k^T)$ is orthogonal with any feasible solution of (4.21), meaning that the trust region constraint in (4.21) is equivalent to $\|n_k^R + t^N\| \leq \delta_k^s$. Thus, as (4.12) are the optimality conditions of (4.8), the optimality conditions of problem (4.21) (with this modified trust region constraint) are that there exists $(t_k^N, y_k^N, \lambda_k^N) \in \mathbb{R}^N \times \mathbb{R}^M \times \mathbb{R}$ such that

$$\begin{bmatrix} H_k + \lambda_k^N I & J_k^T \\ J_k & 0 \end{bmatrix} \begin{bmatrix} t_k^N \\ y_k^N \end{bmatrix} = - \begin{bmatrix} g_k + (H_k + \lambda_k^N I) n_k^R \\ 0 \end{bmatrix}, \tag{4.23a}$$

$$Z_k^T H_k Z_k + \lambda_k^N I \succeq 0, \tag{4.23b}$$

$$\text{and} \quad \lambda_k^N \perp (\delta_k^s - \|n_k^R + t_k^N\|) \geq 0. \tag{4.23c}$$

From equivalence of the systems (4.23) and (4.12), it is clear that $(t_k^N, y_k^N, \lambda_k^N)$ is a primal-dual solution of (4.21) (with the modified trust region constraint) if and only if $(t_k, y_k^f, \lambda_k^f) = (t_k^N - n_k^N, y_k^N, \lambda_k^N)$ is a primal-dual solution of (4.8). This proves part (i).

Part (ii) follows in a similar manner from the orthogonal decomposition $n_k = n_k^N + n_k^R$ and the fact that $J_k t^N = 0$ in (4.22) ensures that $t_k^N \in \text{Null}(J_k)$. □

The next lemma reveals important properties of the tangential step. In particular, it shows that the procedure for performing a contraction of the trust region radius in an F-ITERATION that results in a rejected step is well-defined.

Lemma 45. *If $k \in \mathcal{C}^f$ and the condition in Step 12 of Algorithm 9 tests true, then there exists $\lambda^f > \lambda_k^f$ such that $\underline{\sigma} \leq \lambda^f / \|n_k + t(\lambda^f)\|$, where $t(\lambda^f)$ solves $\mathcal{Q}_k^f(\lambda^f)$.*

Proof. Since the condition in Step 12 of Algorithm 9 is assumed to test true, it follows that $\lambda_k^f < \underline{\sigma}\|s_k\|$. Second, letting $t(\lambda^f)$ denote the solution of $\mathcal{Q}_k^f(\lambda^f)$, it follows by Lemma 44(ii) that $\lim_{\lambda^f \to \infty} \|n_k + t(\lambda^f)\| = \|n_k^R\|$, meaning that $\lim_{\lambda^f \to \infty} \lambda^f / \|n_k + t(\lambda^f)\| = \infty$. It follows from these observations and standard theory for trust region methods [23, Chapter 7] that the result is true. □

The next lemma reveals properties of the normal step trust region radii along with some additional observations about the sequences $\{\Delta_k^v\}$, $\{\lambda_k^v\}$, and $\{\sigma_k^v\}$.

Lemma 46. *The following hold:*

(i) if $k \in \mathcal{C}^v$, then $0 < \delta_{k+1}^v < \delta_k^v$ and $\lambda_{k+1}^v \geq \lambda_k^v$;

(ii) if $k \in \mathcal{I}$, then $\delta_k^v \leq \Delta_k^v \leq \Delta_{k+1}^v$;

(iii) if $k \in \mathcal{S}^v \cup \mathcal{E}^v$, then $\delta_{k+1}^v \geq \delta_k^v$; and

(iv) if $k \in \mathcal{F}$, then $\delta_{k+1}^v = \delta_k^v$ and $\sigma_{k+1}^v = \sigma_k^v$.

Proof. The proof of part (i) follows as that of Lemma 3. In particular, since the V-CONTRACT procedure follows exactly that of CONTRACT in Algorithm 6, it follows that any call of V-CONTRACT results in a contraction of the trust region radius for the normal subproblem and non-decrease of the corresponding dual variable.

For part (ii), the result is trivial if $\mathcal{I} = \emptyset$. Thus, let us assume that $\mathcal{I} \neq \emptyset$, which ensures that $0 \in \mathcal{I}$. We now first prove $\delta_k^v \leq \Delta_k^v$ for all $k \in \mathcal{I}$ using induction. By the initialization procedure of Algorithm 8, it follows that $\delta_0^v \leq \Delta_0^v$. Hence, let us proceed by assuming that $\delta_k^v \leq \Delta_k^v$ for some $k \in \mathcal{I}$. If $k \in \mathcal{S}^v$, then Step 6 of Algorithm 10 shows that $\delta_{k+1}^v \leq \Delta_{k+1}^v$. If $k \in \mathcal{E}^v$, then Step 16 of Algorithm 10 gives $\delta_{k+1}^v \leq \Delta_{k+1}^v$.

If $k \in \mathcal{C}^v$, then part (i), Step 10 of Algorithm 10, and the induction hypothesis yield $\delta_{k+1}^v < \delta_k^v \leq \Delta_k^v = \Delta_{k+1}^v$. Lastly, if $k \in \mathcal{F}$, then Step 12 of Algorithm 8 and the inductive hypothesis give $\delta_{k+1}^v = \delta_k^v \leq \Delta_k^v = \Delta_{k+1}^v$. The induction step has now been completed since we have overall proved that $\delta_{k+1}^v \leq \Delta_{k+1}^v$, which means that we have proved the first inequality in part (ii). To prove $\Delta_k^v \leq \Delta_{k+1}^v$, consider two cases. If $k \in \mathcal{S}^v$, then Step 5 of Algorithm 10 gives $\Delta_{k+1}^v \geq \Delta_k^v$. Otherwise, if $k \notin \mathcal{S}^v$, then according to Step 12 of Algorithm 8 and Steps 10 and 15 of Algorithm 10, it follows that $\Delta_{k+1}^v = \Delta_k^v$. Combining both cases, the proof of part (ii) is now complete.

For part (iii), first observe from part (ii) and Step 6 of Algorithm 10 that if $k \in \mathcal{S}^v$, then $\delta_{k+1}^v = \min\{\Delta_{k+1}^v, \max\{\delta_k^v, \gamma_{\text{E}}\|n_k\|\}\} \geq \delta_k^v$. On the other hand, if $k \in \mathcal{E}^v$, then the conditions that must hold true for Step 12 of Algorithm 10 to be reached ensure that $\lambda_k^v > 0$, meaning that $\|n_k\| = \delta_k^v$ (see (4.6c)). From this and the fact that the conditions in Step 12 of Algorithm 10 must hold true, it follows that $\lambda_k^v/\sigma_k^v > \|n_k\| = \delta_k^v$ and $\|n_k\| < \Delta_k^v$. Combining these observations with $\Delta_{k+1}^v = \Delta_k^v$ for $k \in \mathcal{E}^v$ (see Step 15 of Algorithm 10) it follows from Step 16 of Algorithm 10 that $\delta_{k+1}^v > \|n_k\| = \delta_k^v$.

Finally, part (iv) follows from Steps 12 and 26 of Algorithm 8. □

The next result reveals similar properties for the other radii and $\{\lambda_k^f\}$.

Lemma 47. *The following hold:*

(i) if $k \in \mathcal{C}^f$, then $\delta_{k+1}^f < \delta_k^f$ and if, in addition, $(k+1) \in \mathcal{I}^t$, then $\lambda_{k+1}^f \geq \lambda_k^f$;

(ii) if $k \in \mathcal{S}^f$, then $\delta_{k+1}^f \geq \delta_k^f$ and $\delta_{k+1}^s \geq \delta_k^s$; and

(iii) if $k \in \mathcal{V}$, then $\delta_{k+1}^f = \delta_k^f$.

Proof. For part (i), notice that δ_{k+1}^f is set in Step 9 of Algorithm 9 and that $(x_{k+1}, \delta_{k+1}^v) \leftarrow (x_k, \delta_k^v)$ and $n_{k+1} = n_k$ for all $k \in \mathcal{C}^f$. Let us proceed by considering two cases depending on the condition in Step 12 of Algorithm 9.

Case 1: $\lambda_k^f < \underline{\sigma}\|s_k\|$. In this case, δ_{k+1}^f is set in Step 14 of Algorithm 9, which from Step 13 of Algorithm 9 and Lemma 45 implies that $\lambda^f > \lambda_k^f$. Combining this with Lemma 44 and standard theory for trust region methods leads to the fact that the solution $t^N(\lambda^f)$ of $\bar{\mathcal{Q}}_k^f(\lambda^f)$ satisfies $\|t^N(\lambda^f)\| < \|t_k^N\|$. Thus, $\delta_{k+1}^f = \|n_k + t(\lambda^f)\| = \|n_k^R + t^N(\lambda^f)\| < \|n_k^R + t_k^N\| = \|s_k\| \leq \delta_k^f$, where the last inequality comes from (4.10). If, in addition, $(k+1) \in \mathcal{I}^t$ so that a nonzero tangential step is computed and not reset

to zero, it follows that $\lambda^f_{k+1} = \lambda^f$. This establishes the last conclusion of part (i) for this case since it has already been shown above that $\lambda^f > \lambda^f_k$.

Case 2: $\lambda^f_k \geq \underline{\sigma}\|s_k\|$. In this case, δ^f_{k+1} is set in Step 16 of Algorithm 9 and, from (4.10) and $\gamma_{\rm C} \in (0,1)$, it follows that $\delta^f_{k+1} = \gamma_{\rm C}\|s_k\| \leq \gamma_{\rm C}\delta^f_k < \delta^f_k$. Consequently, from Step 12 of Algorithm 8 and (4.10), one finds that $\delta^s_{k+1} \leq \delta^s_k$. It then follows from Lemma 44 and standard trust region theory that if $(k+1) \in \mathcal{I}^t$, then $\lambda^f_{k+1} \geq \lambda^f_k$.

To prove part (ii), notice that for $k \in \mathcal{S}^f$ it follows by Step 5 of Algorithm 9 that $\delta^f_{k+1} = \max\{\delta^f_k, \gamma_{\rm E}\|s_k\|\}$, so $\delta^f_{k+1} \geq \delta^f_k$. From this, Step 12 of Algorithm 8, and (4.10) it follows that $\delta^s_{k+1} \geq \delta^s_k$. These conclusions complete the proof of part (ii).

Finally, part (iii) follows from Step 16 of Algorithm 8. □

Next, we show that after a V-ITERATION with either a contraction or an expansion of the trust region radius, the subsequent iteration cannot result in an expansion.

Lemma 48. *If $k \in \mathcal{C}^v \cup \mathcal{E}^v$, then $(k+1) \in \mathcal{F} \cup \mathcal{S}^v \cup \mathcal{C}^v$.*

Proof. If $(k+1) \in \mathcal{F}$, then there is nothing left to prove. Otherwise, if $(k+1) \in \mathcal{V}$, then the proof follows using the same logic as for Lemma 6, which shows that one of three cases holds: *(i)* $k \in \mathcal{C}^v$, which yields $\lambda^v_{k+1} \leq \sigma^v_{k+1}\|n_{k+1}\|$, so $(k+1) \notin \mathcal{E}^v$; *(ii)* $k \in \mathcal{E}^v$ and $\Delta^v_k \geq \lambda^v_k/\sigma^v_k$, which also yields $\lambda^v_{k+1} \leq \sigma^v_{k+1}\|n_{k+1}\|$, so $(k+1) \notin \mathcal{E}^v$; or *(iii)* $k \in \mathcal{E}^v$ and $\Delta^v_k < \lambda^v_k/\sigma^v_k$, which implies $(k+1) \in \mathcal{S}^v \cup \mathcal{C}^v$, so $(k+1) \notin \mathcal{E}^v$. □

Our goal now is to expand upon the conclusions of Lemma 48. To do this, it will be convenient to define the first index in a given index set following an earlier index $\bar{k} \in \mathcal{I}$ in that index set (or the initial index 0). In particular, let us define

$$k_{\mathcal{S}}(\bar{k}) := \min\{k \in \mathcal{S} : k > \bar{k}\} \quad \text{and} \quad k_{\mathcal{S}\cup\mathcal{V}}(\bar{k}) := \min\{k \in \mathcal{S}\cup\mathcal{V} : k > \bar{k}\}$$

along with the associated sets

$$\mathcal{I}_{\mathcal{S}}(\bar{k}) := \{k \in \mathcal{I} : \bar{k} < k < k_{\mathcal{S}}(\bar{k})\} \quad \text{and} \quad \mathcal{I}_{\mathcal{S}\cup\mathcal{V}}(\bar{k}) := \{k \in \mathcal{I} : \bar{k} < k < k_{\mathcal{S}\cup\mathcal{V}}(\bar{k})\}.$$

The following lemma shows one important property related to these quantities.

Lemma 49. *For all $\bar{k} \in \mathcal{S} \cup \{0\}$, it follows that $|\mathcal{E}^v \cap \mathcal{I}_{\mathcal{S}}(\bar{k})| \leq 1$.*

Proof. In order to derive a contradiction, suppose that there exists $\bar{k} \in \mathcal{S} \cup \{0\}$ such that $|\mathcal{E}^v \cap \mathcal{I}_{\mathcal{S}}(\bar{k})| > 1$, which means that one can choose $k_{\mathcal{S}_1}$ and $k_{\mathcal{S}_3}$ as the first two distinct indices in $\mathcal{E}^v \cap \mathcal{I}_{\mathcal{S}}(\bar{k})$; in particular,

$$\{k_{\mathcal{S}_1}, k_{\mathcal{S}_3}\} \subseteq \mathcal{E}^v \cap \mathcal{I}_{\mathcal{S}}(\bar{k}) \quad \text{and} \quad \bar{k} < k_{\mathcal{S}_1} < k_{\mathcal{S}_3} < k_{\mathcal{S}}(\bar{k}).$$

By Lemma 48 and the fact that $k_{\mathcal{S}_1} \in \mathcal{E}^v$, it follows that $\{k_{\mathcal{S}_1}+1, \ldots, k_{\mathcal{S}_3}-1\} \neq \emptyset$. Let us proceed by considering two cases, deriving a contradiction in each case.

Case 1: $\mathcal{V} \cap \{k_{\mathcal{S}_1}+1, \ldots, k_{\mathcal{S}_3}-1\} = \emptyset$. In this case, by the definitions of $k_{\mathcal{S}_1}$, $k_{\mathcal{S}_3}$, and $\mathcal{I}_{\mathcal{S}}(\bar{k})$, it follows that $\{k_{\mathcal{S}_1}+1, \ldots k_{\mathcal{S}_3}-1\} \subseteq \mathcal{C}^f$. Then, since $\delta^v_{k+1} = \delta^v_k$ and $\sigma^v_{k+1} = \sigma^v_k$ for all $k \in \mathcal{C}^f \subseteq \mathcal{F}$, it follows that $\delta^v_{k_{\mathcal{S}_3}} = \delta^v_{k_{\mathcal{S}_1}+1}$ and $\sigma^v_{k_{\mathcal{S}_3}} = \sigma^v_{k_{\mathcal{S}_1}+1}$. In particular, using the fact that $\delta^v_{k_{\mathcal{S}_3}} = \delta^v_{k_{\mathcal{S}_1}+1}$, it follows along with the fact that $x_{k+1} = x_k$ for all $k \notin \mathcal{S}$ that $\|n_{k_{\mathcal{S}_3}}\| = \|n_{k_{\mathcal{S}_1}+1}\|$ and $\lambda^v_{k_{\mathcal{S}_3}} = \lambda^v_{k_{\mathcal{S}_1}+1}$. Now, since $(k_{\mathcal{S}_1}+1) \in \mathcal{C}^f$, it follows with Step 9 of Algorithm 8 and (4.15e) that

$$\lambda^v_{k_{\mathcal{S}_3}}/\|n_{k_{\mathcal{S}_3}}\| = \lambda^v_{k_{\mathcal{S}_1}+1}/\|n_{k_{\mathcal{S}_1}+1}\| \leq \sigma^v_{k_{\mathcal{S}_1}+1} = \sigma^v_{k_{\mathcal{S}_3}},$$

which implies that $k_{\mathcal{S}_3} \notin \mathcal{E}^v$, a contradiction.

Case 2: $\mathcal{V} \cap \{k_{\mathcal{S}_1}+1, \ldots, k_{\mathcal{S}_3}-1\} \neq \emptyset$. In this case, by the definitions of $k_{\mathcal{S}_1}$, $k_{\mathcal{S}_3}$, and $\mathcal{I}_{\mathcal{S}}(\bar{k})$, it follows that $\{k_{\mathcal{S}_1}+1, \ldots, k_{\mathcal{S}_3}-1\} \subseteq \mathcal{C}^f \cup \mathcal{C}^v$. In addition, by the condition of this case, it also follows that there exists a greatest index $k_{\mathcal{S}_2} \in \mathcal{C}^v \cap \{k_{\mathcal{S}_1}+1, \ldots, k_{\mathcal{S}_3}-1\}$. In particular, for the index $k_{\mathcal{S}_2} \in \mathcal{C}^v$, it follows that $k_{\mathcal{S}_1}+1 \leq k_{\mathcal{S}_2} \leq k_{\mathcal{S}_3}-1$ and $\{k_{\mathcal{S}_2}+1, \ldots, k_{\mathcal{S}_3}-1\} \subseteq \mathcal{C}^f$. By $k_{\mathcal{S}_2} \in \mathcal{C}^v$ and Lemma 48, it follows that $k_{\mathcal{S}_2}+1 \notin \mathcal{E}^v$; hence, since $k_{\mathcal{S}_3} \in \mathcal{E}^v$, it follows that $\{k_{\mathcal{S}_2}+1, \ldots, k_{\mathcal{S}_3}-1\} \neq \emptyset$. We may now apply the same argument as for Case 1, but with $k_{\mathcal{S}_1}$ replaced by $k_{\mathcal{S}_2}$, to arrive at the contradictory conclusion that $k_{\mathcal{S}_3} \notin \mathcal{E}^v$, completing the proof. □

The next lemma reveals lower bounds for the norms of the normal and full steps.

Lemma 50. *For all $k \in \mathcal{I}$, the following hold:*

(i) $\|n_k\| \geq \min\{\delta^v_k, \|g^v_k\|/\|H^v_k\|\} > 0$ *and*

(ii) $\|s_k\| \geq \kappa_{ntn} \min\{\delta^v_k, \|g^v_k\|/\|H^v_k\|\} > 0.$

Proof. The proof of part (i) follows as that for Lemma 1. Part (ii) follows from part (i) and (4.14b), the latter of which holds because of Lemma 41(ii). □

We now provide a lower bound for the decrease in the model of infeasibility.

Lemma 51. *For all $k \in \mathcal{I}$, the quantities n_k, λ_k^v, and s_k satisfy*

$$v_k - m_k^v(n_k) = \tfrac{1}{2} n_k^T (H_k^v + \lambda_k^v I) n_k + \tfrac{1}{2}\lambda_k^v \|n_k\|^2 > 0, \tag{4.24a}$$

$$v_k - m_k^v(s_k) \geq \kappa_{vm}(\tfrac{1}{2} n_k^T (H_k^v + \lambda_k^v I) n_k + \tfrac{1}{2}\lambda_k^v \|n_k\|^2) > 0, \quad \textit{and} \tag{4.24b}$$

$$v_k - m_k^v(s_k) \geq \tfrac{1}{2}\kappa_{vm}\|g_k^v\| \min\{\delta_k^v, \|g_k^v\|/\|H_k^v\|\} > 0. \tag{4.24c}$$

Proof. The proof of (4.24a) follows as for that of Lemma 2 and the fact that $\|g_k^v\| > \varepsilon$, which holds since $k \in \mathcal{I}$. The inequalities in (4.24b) follow from (4.24a) and (4.14a), the latter of which holds because of Lemma 41(ii). To prove (4.24c), first observe from standard trust region theory (e.g., see [23, Theorem 6.3.1]) that

$$v_k - m_k^v(n_k) \geq \tfrac{1}{2}\|g_k^v\| \min\{\delta_k^v, \|g_k^v\|/\|H_k^v\|\} > 0. \tag{4.25}$$

By combining (4.25) and (4.14a) (which holds by Lemma 41(ii)), one obtains (4.24c). □

The next lemma reveals that if the dual variable for the normal step trust region is beyond a certain threshold, then the trust region constraint must be active and the step will either be an F-ITERATION or a successful V-ITERATION.

Lemma 52. *For all $k \in \mathcal{I}$, if the trial step s_k and the dual variable λ_k^v satisfy*

$$\lambda_k^v \geq \kappa_\delta^2(2g_{Lip}^v + \theta_{fc} + 2\kappa_\rho\|s_k\|)/\kappa_{vm}, \tag{4.26}$$

then $\|n_k\| = \delta_k^v$ and $\rho_k^v \geq \kappa_\rho$.

Proof. For all $k \in \mathcal{I}$, it follows from the definition of m_k^v and the Mean Value Theorem that there exists a point $\bar{x}_k \in \mathbb{R}^N$ on the line segment $[x_k, x_k + s_k]$ such that

$$\begin{aligned} m_k^v(s_k) - v(x_k + s_k) &= (g_k^v - g^v(\bar{x}_k))^T s_k + \tfrac{1}{2} s_k^T H_k^v s_k \\ &\geq -\|g_k^v - g^v(\bar{x}_k)\|\|s_k\| - \tfrac{1}{2}\|H_k^v\|\|s_k\|^2. \end{aligned} \tag{4.27}$$

By (4.26) and (4.6c), it follows that $\|n_k\| = \delta_k^v$. Combining this fact with (4.27), (4.24b),

(4.6b), Lemma 40, (4.26), and the fact that $\|s_k\| \leq \delta_k^s \leq \kappa_\delta \delta_k^v = \kappa_\delta \|n_k\|$, one obtains

$$\begin{aligned} v_k - v(x_k + s_k) &= v_k - m_k^v(s_k) + m_k^v(s_k) - v(x_k + s_k) \\ &\geq \tfrac{1}{2}\kappa_{vm}\lambda_k^v \|n_k\|^2 - \|g_k^v - g^v(\bar{x}_k)\|\|s_k\| - \tfrac{1}{2}\|H_k^v\|\|s_k\|^2 \\ &\geq (\tfrac{1}{2}\kappa_{vm}\kappa_\delta^{-2}\lambda_k^v - g_{Lip}^v - \tfrac{1}{2}\theta_{fc})\|s_k\|^2 \geq \kappa_\rho \|s_k\|^3, \end{aligned}$$

which, by Steps 12 and 14 in Algorithm 8 and (4.18), completes the proof. □

Recall that our main goal in this section is to prove that $|\mathcal{I}| < \infty$. Ultimately, this result is attained by deriving contradictions under the assumption that $|\mathcal{I}| = \infty$. For example, if $|\mathcal{I}| = \infty$ and the iterations corresponding to all sufficiently large $k \in \mathcal{I}$ involve contractions of a trust region radius, then the following lemma helps to lead to contradictions in subsequent results. In particular, it reveals that, under these conditions, a corresponding dual variable tends to infinity.

Lemma 53. *The following hold:*

(i) If $k \notin \mathcal{S}$ for all large $k \in \mathcal{I}$ and $|\mathcal{C}^v| = \infty$, then $\{\delta_k^v\} \to 0$ and $\{\lambda_k^v\} \to \infty$.

(ii) If $k \in \mathcal{C}^f$ for all large $k \in \mathcal{I}$, then $\{\delta_k^f\} \to 0$ and $\{\lambda_k^f\} \to \infty$.

Proof. By Lemma 46, Lemma 49, and the fact that $k \notin \mathcal{S}$ for all large $k \in \mathcal{I}$, the proof of part (i) follows as that of Lemma 8.

To prove part (ii), let us assume, without loss of generality, that $k \in \mathcal{C}^f$ for all $k \in \mathcal{I}$. It then follows that $k \in \mathcal{I}^t$ for all $k \in \mathcal{I}$, since otherwise it would follow that $t_k \leftarrow 0$, which by (4.15a) means $k \in \mathcal{V}$, a contradiction to $k \in \mathcal{C}^f$. Thus,

$$k \in \mathcal{C}^f \cap \mathcal{I}^t \quad \text{for all} \quad k \in \mathcal{I}. \tag{4.28}$$

Next, we claim that the condition in Step 12 of Algorithm 9 can hold true for at most one iteration. If it never holds true, then there is nothing left to prove. Otherwise, let $k_c \in \mathcal{I}$ be the first index for which the condition holds true. The structure of Algorithm 9 (see Step 13) and (4.28) then ensure that $\lambda_{k_c+1}^f / \|s_{k_c+1}\| \geq \underline{\sigma}$. From Lemma 47(i), one may conclude that $\{\lambda_k^f / \|s_k\|\}$ is nondecreasing. From this, it follows that the condition in Step 12 of Algorithm 9 will never be true for any $k > k_c$. Thus, we may now proceed, without loss of generality, under the assumption that the condition in Step 12 of

Algorithm 9 always tests false. This means that δ_{k+1}^f is set in Step 16 of Algorithm 9 for all $k \in \mathcal{I}$, yielding $\delta_{k+1}^f \leftarrow \gamma_C \|s_k\| \leq \gamma_C \delta_k^f$, where the last inequality comes from (4.10). Therefore, $\{\delta_k^f\} \to 0$ for all $k \in \mathcal{I}$, and consequently $\{\lambda_k^f\} \to \infty$. □

We now show that the sequences $\{\Delta_k^v\}$ and $\{n_k\}$ are bounded above.

Lemma 54. *There exists a scalar $\Delta_{\max}^v \in (0, \infty)$ such that $\Delta_k^v = \Delta_{\max}^v$ for all sufficiently large $k \in \mathcal{I}$. In addition, $|\mathcal{S}_\Delta^v| < \infty$ and there exists a scalar $n_{\max} \in (0, \infty)$ such that $\|n_k\| \leq n_{\max}$ for all $k \in \mathcal{I}$.*

Proof. First, in order to derive a contradiction, assume that there is no $\Delta_{\max}^v$ such that $\Delta_k^v = \Delta_{\max}^v$ for all sufficiently large $k \in \mathcal{I}$. This, in turn, means that Step 5 of Algorithm 10 is reached infinitely often, meaning that $|\mathcal{S}^v| = \infty$. For all $k \in \mathcal{S}^v \subseteq \mathcal{S}$, it follows from Lemma 43 that $v_k^{\max} - v_{k+1}^{\max} \geq \kappa_\rho(1 - \kappa_{v2})\|s_k\|^3$. Now, using the monotonicity of $\{v_k^{\max}\}$ and the fact that $v_k^{\max} \geq 0$ (see Lemma 42), one may conclude that $\{v_k^{\max}\}$ converges; therefore $\{s_k\}_{k \in \mathcal{S}^v} \to 0$. From this fact, Lemma 41(ii), and (4.14b) it follows that $\{n_k\}_{k \in \mathcal{S}^v} \to 0$. Thus, there exists an iteration index k_Δ^v such that for all $k \in \mathcal{S}^v$ with $k \geq k_\Delta^v$, one finds $\gamma_E \|n_k\| < \Delta_0^v \leq \Delta_k^v$, where the last inequality follows from Lemma 46(ii). From this and Steps 5, 10, and 15 of Algorithm 10, it follows that $\Delta_{k+1}^v \leftarrow \Delta_k^v$ for all $k \geq k_\Delta^v$, a contradiction. The proof of the second part of the lemma follows as in that for Lemma 10. □

In the next lemma, a uniform lower bound on $\{\delta_k^v\}$ is provided.

Lemma 55. *There exists a scalar $\delta_{\min}^v \in (0, \infty)$ such that $\delta_k^v \geq \delta_{\min}^v$ for all $k \in \mathcal{I}$.*

Proof. If $|\mathcal{C}^v| < \infty$, then the result follows from Lemma 46(iii)–(iv). Thus, let us proceed under the assumption that $|\mathcal{C}^v| = \infty$. As in the beginning of the proof of Lemma 52, it follows that (4.27) holds. Then, using (4.27), (4.24c), Lemma 40(i), Lemma 40(iii), $\|g_k^v\| > \epsilon$ for $k \in \mathcal{I}$, and $\|s_k\| \leq \delta_k^s \leq \kappa_\delta \delta_k^v$, it follows that

$$\begin{aligned} v_k - v(x_k + s_k) &= v_k - m_k^v(s_k) + m_k^v(s_k) - v(x_k + s_k) \\ &\geq \tfrac{1}{2}\kappa_{vm}\epsilon \min\{\delta_k^v, \epsilon/\theta_{fc}\} - (g_{Lip}^v + \tfrac{1}{2}\theta_{fc})\kappa_\delta^2(\delta_k^v)^2. \end{aligned}$$

Considering these inequalities and $\|s_k\| \leq \delta_k^s \leq \kappa_\delta \delta_k^v$, it must hold that $\rho_k^v \geq \kappa_\rho$ for any

$k \in \mathcal{I}$ as long as $\delta_k^v \in (0, \epsilon/\theta_{fc}]$ is sufficiently small such that

$$\tfrac{1}{2}\kappa_{vm}\epsilon\delta_k^v - (g_{Lip}^v + \tfrac{1}{2}\theta_{fc})\kappa_\delta^2(\delta_k^v)^2 \geq \kappa_\rho\kappa_\delta^3(\delta_k^v)^3 \geq \kappa_\rho\|s_k\|^3.$$

This fact implies the existence of a positive threshold $\delta_{thresh}^v \in (0, \epsilon/\theta_{fc}]$ such that, for any $k \in \mathcal{I}$ with $\delta_k^v \in (0, \delta_{thresh}^v)$, one finds $\rho_k^v \geq \kappa_\rho$. Along with the fact that $\rho_k^v < \kappa_\rho$ if and only if $k \in \mathcal{C}^v$ (see Step 2, 7, and 12 of Algorithm 10 and Step 12 of Algorithm 8), it follows that

$$\delta_k^v \geq \delta_{thresh}^v \text{ for all } k \in \mathcal{C}^v. \tag{4.29}$$

Since the normal step subproblem trust region radius is only decreased when $k \in \mathcal{C}^v$, we will complete the proof by showing a lower bound on δ_{k+1}^v when $k \in \mathcal{C}^v$.

Suppose that $k \in \mathcal{C}^v$. If Step 24 of Algorithm 10 is reached, then

$$\delta_{k+1}^v \leftarrow \|n(\lambda^v)\| \geq \lambda^v/\overline{\sigma} = (\lambda_k^v + (\underline{\sigma}\|g_k^v\|)^{1/2})/\overline{\sigma} \geq (\underline{\sigma}\|g_k^v\|)^{1/2}/\overline{\sigma} \geq (\underline{\sigma}\epsilon)^{1/2}/\overline{\sigma},$$

where the last inequality follows since $k \in \mathcal{I}$ means $\|g_k^v\| \geq \epsilon$. If Step 27 is reached, then the algorithm chooses $\lambda^v \in (\lambda_k^v, \hat{\lambda}^v)$ to find $n(\lambda^v)$ that solves $\mathcal{Q}_k^v(\lambda^v)$ such that $\underline{\sigma} \leq \lambda^v/\|n(\lambda^v)\| \leq \overline{\sigma}$. For this case and the cases when Step 32 or 34 is reached, the existence of $\delta_{\min}^v \in (0, \infty)$ such that $\delta_{k+1}^v \geq \delta_{\min}^v$ for all $k \in \mathcal{C}^v$ follows in the same manner as in the proof of Lemma 11. Combining these facts with (4.29) and Lemma 46(iii)–(iv), the proof is complete. □

The next result shows that there are finitely many successful iterations.

Lemma 56. *The following hold:* $|\mathcal{S}^v| < \infty$ *and* $|\mathcal{S}^f| < \infty$.

Proof. Lemma 55, $\|g_k^v\| > \epsilon$ for all $k \in \mathcal{I}$, Lemma 50(i), and Lemma 40(i) imply the existence of $n_{\min} \in (0, \infty)$ such that $\|n_k\| \geq n_{\min}$ for all $k \in \mathcal{I}$, i.e.,

$$\|g_k^v\| > \epsilon \quad \text{and} \quad \|n_k\| \geq n_{\min} > 0 \quad \text{for all } k \in \mathcal{I}. \tag{4.30}$$

In order to reach a contradiction to the first desired conclusion, suppose that $|\mathcal{S}^v| = \infty$. For any $k \in \mathcal{S}^v$, it follows from Lemma 43, Lemma 41(ii), and (4.14b) that

$$v_k^{\max} - v_{k+1}^{\max} \geq \kappa_\rho(1 - \kappa_{v2})\|s_k\|^3 \geq \kappa_\rho(1 - \kappa_{v2})\kappa_{ntn}^3\|n_k\|^3. \tag{4.31}$$

By Lemma 42, $0 < v_{k+1}^{\max} \leq v_k^{\max}$ for all $k \in \mathcal{I}$, meaning that $\{v_k^{\max} - v_{k+1}^{\max}\} \to 0$, which together with (4.31) shows that $\{\|n_k\|\}_{k\in\mathcal{S}^v} \to 0$, contradicting (4.30). This proves that $|\mathcal{S}^v| < \infty$. Now, in order to reach a contradiction to the second desired conclusion, suppose that $|\mathcal{S}^f| = \infty$. Since $|\mathcal{S}^v| < \infty$, we can assume without loss of generality that $\mathcal{S} = \mathcal{S}^f$. This means that the sequence $\{f_k\}$ is monotonically nonincreasing. Combining this with the fact that $\{f_k\}$ is bounded below under Assumptions 12 and 13, it follows that $\{f_k\} \to f_{low}$ for some $f_{low} \in (-\infty, \infty)$ and $\{f_k - f_{k+1}\} \to 0$. Using these facts, the inequality $\rho_k^f \geq \kappa_\rho$ for all $k \in \mathcal{S}^f$, and $|\mathcal{S}^f| = \infty$, it follows that $\{\kappa_\rho \|s_k\|^3\}_{k\in\mathcal{S}^f} \leq \{f_k - f_{k+1}\}_{k\in\mathcal{S}^f} \to 0$, which gives $\{\|s_k\|\}_{k\in\mathcal{S}^f} \to 0$. This, in turn, implies that $\{\|n_k\|\}_{k\in\mathcal{S}^f} \to 0$ because of Lemma 41(ii) and (4.14b), which contradicts (4.30). Hence, $|\mathcal{S}^f| < \infty$. □

We are now prepared to prove that Algorithm 8 terminates finitely.

Theorem 12. *Algorithm 8 terminates finitely, i.e., $|\mathcal{I}| < \infty$.*

Proof. Suppose by contradiction that $|\mathcal{I}| = \infty$. Let us consider two cases.

Case 1: $|\mathcal{V}| = \infty$. Since $|\mathcal{S}| < \infty$, it follows that $|\mathcal{V} \setminus \mathcal{S}^v| = |\mathcal{C}^v \cup \mathcal{E}^v| = \infty$, which along with Lemma 49 implies that $|\mathcal{E}^v| < \infty$ while $|\mathcal{C}^v| = \infty$. It now follows from Lemma 53(i) that $\{\delta_k^v\} \to 0$, which contradicts Lemma 55.

Case 2: $|\mathcal{V}| < \infty$. For this case, we may assume without loss of generality that $\mathcal{F} = \mathcal{I}$. This implies with Lemma 41(i) that $\delta_k^v = \delta_0^v$ and $n_k = n_0 \neq 0$ for all $k \in \mathcal{I}$. It also implies from Step 9 of Algorithm 8 that (4.15) holds for all $k \in \mathcal{I}$; in particular, from (4.15a) it means that $t_k \neq 0$ for all $k \in \mathcal{I}$. Now, from $|\mathcal{V}| < \infty$, $|\mathcal{S}| < \infty$, and Lemma 53(ii), it follows that $\{\delta_k^f\} \to 0$, which by (4.10) yields $\{\delta_k^s\} \to 0$. It then follows from Step 20 of Algorithm 8 and $\mathcal{F} = \mathcal{I}$ that $\{n_k\} \to 0$, which contradicts our previous conclusion that $n_k = n_0 \neq 0$ for all $k \in \mathcal{I}$. □

4.3.2 Complexity analysis for phase 1

Our goal in this subsection is to prove an upper bound on the total number of iterations required until phase 1 terminates, i.e., until the algorithm reaches $k \in \mathbb{N}$ such that $\|g_k^v\| \leq \epsilon$. To prove such a bound, we require the following additional assumption.

Assumption 14. *The Hessian functions $H^v(x) := \nabla^2 v(x)$ and $\nabla^2 f(x)$ are Lipschitz continuous with constants $H_{Lip}^v \in (0, \infty)$ and $H_{Lip} \in (0, \infty)$, respectively, on a path defined by the sequence of iterates and trial steps computed in the algorithm.*

Our first result in this subsection can be seen as a similar conclusion to that given by Lemma 52, but with this additional assumption in hand.

Lemma 57. *For all $k \in \mathcal{I}$, if the trial step s_k and dual variable λ_k^v satisfy*

$$\lambda_k^v \geq \kappa_\delta^2 \kappa_{vm}^{-1}(H_{Lip}^v + 2\kappa_\rho)\|s_k\|, \tag{4.32}$$

then $\|n_k\| = \delta_k^v$ and $\rho_k^v \geq \kappa_\rho$.

Proof. For all $k \in \mathcal{I}$, there exists $\overline{x}_k$ on the line segment $[x_k, x_k + s_k]$ such that

$$m_k^v(s_k) - v(x_k + s_k) = \tfrac{1}{2} s_k^T \left(H_k^v - H^v(\overline{x}_k)\right) s_k \geq -\tfrac{1}{2} H_{Lip}^v \|s_k\|^3. \tag{4.33}$$

From this, (4.24b), and (4.6b), one deduces that

$$v(x_k) - v(x_k + s_k) \geq \tfrac{1}{2}\kappa_{vm}\lambda_k^v \|n_k\|^2 - \tfrac{1}{2} H_{Lip}^v \|s_k\|^3.$$

From Lemma 41(i), (4.32), and (4.6c), it follows that $\|n_k\| = \delta_k^v$, which along with (4.10) means that $\|s_k\| \leq \delta_k^s \leq \kappa_\delta \delta_k^v = \kappa_\delta \|n_k\|$, so, from above,

$$v(x_k) - v(x_k + s_k) \geq \tfrac{1}{2}\kappa_{vm}\kappa_\delta^{-2}\lambda_k^v \|s_k\|^2 - \tfrac{1}{2} H_{Lip}^v \|s_k\|^3. \tag{4.34}$$

From here, by Steps 12 and 14 of Algorithm 8 and under (4.32), the result follows. □

The next lemma reveals upper and lower bounds for an important ratio that will hold during the iteration immediately following a V-ITERATION contraction.

Lemma 58. *For all $k \in \mathcal{C}^v$, it follows that*

$$\underline{\sigma} \leq \lambda_{k+1}^v / \|n_{k+1}\| \leq \max\left\{\overline{\sigma}, \gamma_\lambda \gamma_{\mathrm{C}}^{-1} \lambda_k^v / \|n_k\|\right\}. \tag{4.35}$$

Proof. The result follows using the same logic as the proof of Lemma 14. □

Now, we prove that the sequence $\{\sigma_k^v\}$ is bounded.

Lemma 59. *There exists $\sigma_{\max}^v \in (0, \infty)$ such that $\sigma_k^v \leq \sigma_{\max}^v$ for all $k \in \mathcal{I}$.*

Proof. If $k \notin \mathcal{C}^v$, then Steps 26 and 28 of Algorithm 8 give $\sigma_{k+1}^v \leftarrow \sigma_k^v$. Otherwise, if $k \in \mathcal{C}^v$, meaning that $\rho_k^v < \kappa_\rho$, then there are two cases to consider. If $k \in \mathcal{C}^v$ and

$\lambda_k^v < \underline{\sigma}\|n_k\|$, then, by Steps 23–27 of Algorithm 10, Step 28 of Algorithm 8, and the fact that $\lambda_{k+1}^v = \lambda^v$ and $n_{k+1} = n(\lambda^v)$, where $(n(\lambda^v, \lambda^v)$ are computed either in Steps 21–22 or Step 26 of Algorithm 10, it follows that $\sigma_{k+1}^v \leq \max\{\sigma_k^v, \overline{\sigma}\}$. Finally, if $k \in \mathcal{C}^v$ and $\lambda_k^v \geq \underline{\sigma}\|n_k\|$, then with Lemma 57 one has $\lambda_k^v < \kappa_\delta^2 \kappa_{vm}^{-1}(H_{Lip}^v + 2\kappa_\rho)\|s_k\|$. From the fact that $\lambda_k^v \geq \underline{\sigma}\|n_k\|$, Lemma 41(i), (4.6c), and (4.10), it follows that $\|s_k\| \leq \delta_k^s \leq \kappa_\delta \delta_k^v = \kappa_\delta \|n_k\|$. Hence, by Step 28 of Algorithm 8 and Lemma 58, one finds

$$\sigma_{k+1}^v \leftarrow \max\left\{\sigma_k^v, \lambda_{k+1}^v/\|n_{k+1}\|\right\} \leq \max\left\{\sigma_k^v, \overline{\sigma}, \gamma_\lambda \gamma_{\mathrm{C}}^{-1} \kappa_\delta^3 \kappa_{vm}^{-1}(H_{Lip}^v + 2\kappa_\rho)\right\}.$$

Combining the results of these cases gives the desired conclusion. □

We now give a lower-bound for the norm of some types of successful steps.

Lemma 60. *For all $k \in \mathcal{S}_\sigma^v \cup \mathcal{S}^f$, the accepted step s_k satisfies*

$$\|s_k\| \geq (H_{Lip}^v + \kappa_{ht} + \sigma_{\max}^v/\kappa_{ntn}^2)^{-1/2} \|g_{k+1}^v\|^{1/2}. \tag{4.36}$$

Proof. Let $k \in \mathcal{S}_\sigma^v \cup \mathcal{S}^f$. It follows from (4.6a), the Mean Value Theorem, the fact that $s_k = n_k + t_k$, Assumption 14, Lemma 41(ii), and (4.14c) that

$$\|g_{k+1}^v\| = \|g_{k+1}^v - g_k^v - (H_k^v + I\lambda_k^v)n_k\| \leq (H_{Lip}^v + \kappa_{ht})\|s_k\|^2 + \lambda_k^v \|n_k\|^2/\|n_k\|. \tag{4.37}$$

From Step 2 of Algorithm 10 (if $k \in \mathcal{S}_\sigma^v$) and (4.15e) (if $k \in \mathcal{S}^f$), one finds $\lambda_k^v/\|n_k\| \leq \sigma_k^v$. Combining this with (4.37), Lemma 41(ii), (4.14b), and Lemma 59, it follows that

$$\|g_{k+1}^v\| \leq H_{Lip}^v \|s_k\|^2 + \kappa_{ht}\|s_k\|^2 + \sigma_k^v \|n_k\|^2 \leq (H_{Lip}^v + \kappa_{ht} + \sigma_{\max}^v/\kappa_{ntn}^2)\|s_k\|^2,$$

which gives the desired result. □

We now give an iteration complexity result for a subset of successful iterations.

Lemma 61. *For any $\epsilon \in (0, \infty)$, the total number of elements in*

$$\mathcal{K}(\epsilon) := \{k \in \mathcal{I} : k \geq 0 \text{ and } (k-1) \in \mathcal{S}_\sigma^v \cup \mathcal{S}^f\}$$

is at most

$$\left\lfloor \left(\frac{v_0^{\max}}{\kappa_\rho(1-\kappa_{v2})(H_{Lip}^v + \kappa_{ht} + \sigma_{\max}^v/\kappa_{ntn}^2)^{-3/2}} \right) \epsilon^{-3/2} \right\rfloor =: K_\sigma(\epsilon) \geq 0. \tag{4.38}$$

Proof. From Lemma 43 and Lemma 60, it follows that, for all $k \in \mathcal{K}(\epsilon) \subseteq \mathcal{I}$,

$$\begin{aligned} v_{k-1}^{\max} - v_k^{\max} &\geq \kappa_\rho(1-\kappa_{v2})\|s_{k-1}\|^3 \\ &\geq \kappa_\rho(1-\kappa_{v2})(H_{Lip}^v + \kappa_{ht} + \sigma_{\max}^v/\kappa_{ntn}^2)^{-3/2}\epsilon^{3/2}. \end{aligned}$$

In addition, since $|\mathcal{K}(\epsilon)| < \infty$ follows by Theorem 12, the reduction in $v_k^{\max}$ obtained up to the largest index in $\mathcal{K}(\epsilon)$, call it $\bar{k}(\epsilon)$, satisfies

$$\begin{aligned} v_0^{\max} - v_{\bar{k}(\epsilon)}^{\max} &= \sum_{k=1}^{\bar{k}_\epsilon} (v_{k-1}^{\max} - v_k^{\max}) \geq \sum_{k \in \mathcal{K}(\epsilon)} (v_{k-1}^{\max} - v_k^{\max}) \\ &\geq |\mathcal{K}(\epsilon)|\kappa_\rho(1-\kappa_{v2})(H_{Lip}^v + \kappa_{ht} + \sigma_{\max}^v/\kappa_{ntn}^2)^{-3/2}\epsilon^{3/2}. \end{aligned}$$

Rearranging this inequality to yield an upper bound for $|\mathcal{K}(\epsilon)|$ and using the fact that $v_k^{\max} \geq 0$ for all $k \in \mathcal{I}$ (see Lemma 42), the desired result follows. □

In order to bound the total number of successful iterations in $\mathcal{I}$, we also need an upper bound for the cardinality of $\mathcal{S}_\Delta^v$. This is the subject of our next lemma.

Lemma 62. *The cardinality of the set $\mathcal{S}_\Delta^v$ is bounded above by*

$$\left\lfloor \frac{v_0^{\max}}{\kappa_\rho \kappa_{ntn}^3 (1-\kappa_{v2})(\Delta_0^v)^3} \right\rfloor := K_\Delta^v \geq 0. \tag{4.39}$$

Proof. For all $k \in \mathcal{S}_\Delta^v \subseteq \mathcal{S}$, it follows from Lemma 43, Lemma 41(ii), (4.14b), and Lemma 46(ii) that the decrease in the trust funnel radius satisfies

$$\begin{aligned} v_k^{\max} - v_{k+1}^{\max} &\geq \kappa_\rho(1-\kappa_{v2})\|s_k\|^3 \geq \kappa_\rho\kappa_{ntn}^3(1-\kappa_{v2})\|n_k\|^3 \\ &= \kappa_\rho\kappa_{ntn}^3(1-\kappa_{v2})(\Delta_k^v)^3 \geq \kappa_\rho\kappa_{ntn}^3(1-\kappa_{v2})(\Delta_0^v)^3. \end{aligned}$$

Now, using the fact that $\{v_k^{\max}\}$ is bounded below by zero (see Lemma 42), one finds

$$v_0^{\max} \geq \sum_{k \in \mathcal{S}_\Delta^v} (v_k^{\max} - v_{k+1}^{\max}) \geq |\mathcal{S}_\Delta^v|\kappa_\rho\kappa_{ntn}^3(1-\kappa_{v2})(\Delta_0^v)^3,$$

which gives the desired result. □

Having now provided upper bounds for the numbers of successful iterations, we need to bound the number of unsuccessful iterations in $\mathcal{I}$. To this end, first we prove that a critical ratio increases by at least a constant factor after an iteration in $\mathcal{C}^v$.

Lemma 63. *If $k \in \mathcal{C}^v$ and $\lambda_k^v \geq \underline{\sigma}\|n_k\|$, then $\frac{\lambda_{k+1}^v}{\|n_{k+1}\|} \geq \min\left\{\gamma_\lambda, \frac{1}{\gamma_C}\right\} \frac{\lambda_k^v}{\|n_k\|}$.*

Proof. The proof follows the same logic as in Lemma 20. □

We are now able to provide an upper bound on the number of unsuccessful iterations in $\mathcal{C}^v$ that may occur between any two successful iterations.

Lemma 64. *If $\bar{k} \in \mathcal{S} \cup \{0\}$, then*

$$|\mathcal{C}^v \cap \mathcal{I}_{\mathcal{S}}(\bar{k})| \leq 1 + \left\lfloor \frac{1}{\log(\min\{\gamma_\lambda, \gamma_C^{-1}\})} \log\left(\frac{\sigma_{\max}^v}{\underline{\sigma}}\right) \right\rfloor =: K_{\mathcal{C}}^v \geq 0. \tag{4.40}$$

Proof. The result holds trivially if $|\mathcal{C}^v \cap \mathcal{I}_{\mathcal{S}}(\bar{k})| = 0$. Thus, we may proceed under the assumption that $|\mathcal{C}^v \cap \mathcal{I}_{\mathcal{S}}(\bar{k})| \geq 1$. Let $k_{\mathcal{C}^v}$ be the smallest element in $\mathcal{C}^v \cap \mathcal{I}_{\mathcal{S}}(\bar{k})$. It then follows from Lemma 46(i)-(ii), Lemma 48, and Step 12 of Algorithm 8 that for all $k \in \mathcal{I}$ satisfying $k_{\mathcal{C}^v} + 1 \leq k \leq k_{\mathcal{S}}(\bar{k})$ we have

$$\|n_k\| \leq \delta_k^v \leq \delta_{k_{\mathcal{C}^v}+1}^v < \delta_{k_{\mathcal{C}^v}}^v \leq \Delta_{k_{\mathcal{C}^v}}^v \leq \Delta_{k_{\mathcal{S}}(\bar{k})}^v,$$

which for $k = k_{\mathcal{S}}(\bar{k})$ means that $k_{\mathcal{S}}(\bar{k}) \in \mathcal{S}^f \cup \mathcal{S}_\sigma^v$. From Lemma 58, it follows that $\lambda_{k_{\mathcal{C}^v}+1}^v \geq \underline{\sigma}\|n_{k_{\mathcal{C}^v}+1}\|$, which by $k_{\mathcal{S}}(\bar{k}) \in \mathcal{S}^f \cup \mathcal{S}_\sigma^v$, Lemma 59, Lemma 63, (4.15e), Step 2 of Algorithm 10, and the fact that $(n_{k+1}, \lambda_{k+1}^v) = (n_k, \lambda_k^v)$ for any $k \in \mathcal{C}^f$ means

$$\sigma_{\max}^v \geq \sigma_{k_{\mathcal{S}}(\bar{k})}^v \geq \lambda_{k_{\mathcal{S}}(\bar{k})}^v / \|n_{k_{\mathcal{S}}(\bar{k})}\| \geq \left(\min\left\{\gamma_\lambda, \gamma_C^{-1}\right\}\right)^{|\mathcal{C}^v \cap \mathcal{I}_{\mathcal{S}}(\bar{k})|-1} \underline{\sigma},$$

from which the desired result follows. □

For our ultimate complexity result, the main component that remains to prove is a bound on the number of unsuccessful iterations in $\mathcal{C}^f$ between any two successful iterations. To this end, we first need some preliminary results pertaining to the trial step and related quantities during an F-ITERATION. Our first such result pertains to the change in the objective function model yielded by the tangential step.

Lemma 65. *For any $k \in \mathcal{I}$, the vectors n_k and t_k and dual variable λ_k^f satisfy*

$$m_k^f(n_k) - m_k^f(n_k + t_k) = \tfrac{1}{2} t_k^T (H_k + \lambda_k^f I) t_k + \tfrac{1}{2} \lambda_k^f \|t_k\|^2 + \lambda_k^f n_k^T t_k. \tag{4.41}$$

Proof. If $k \notin \mathcal{I}^t$ so that $t_k = 0$ and $\lambda_k^f = 0$ (by the COMPUTE_STEPS subroutine in Algorithm 8), then (4.41) trivially holds. Thus, for the remainder of the proof, let us assume that $k \in \mathcal{I}^t$. It now follows from the definition of m_k^f that

$$\begin{aligned} m_k^f(n_k) - m_k^f(n_k + t_k) = & -(g_k + H_k n_k)^T t_k - \tfrac{1}{2} t_k^T H_k t_k \\ = & -(g_k + (H_k + \lambda_k^f I) n_k + (H_k + \lambda_k^f I) t_k + J_k^T y_k^f)^T t_k \\ & + \tfrac{1}{2} t_k^T (H_k + \lambda_k^f I) t_k + \tfrac{1}{2} \lambda_k^f \|t_k\|^2 + \lambda_k^f n_k^T t_k + (y_k^f)^T J_k t_k \\ = & \tfrac{1}{2} t_k^T (H_k + \lambda_k^f I) t_k + \tfrac{1}{2} \lambda_k^f \|t_k\|^2 + \lambda_k^f n_k^T t_k, \end{aligned}$$

where the last equality follows from (4.12a). □

The next lemma reveals that, for an F-ITERATION, if the dual variable for the tangential step trust region constraint is large enough, then the trust region constraint is active and the iteration will be successful.

Lemma 66. *For all $k \in \mathcal{F}$, if the trial step s_k and the dual variable λ_k^f satisfy*

$$\lambda_k^f \geq (\kappa_{fm} \kappa_{st}^2 (1 - \kappa_{ntt}))^{-1} (\kappa_{hs} + H_{Lip} + 2\kappa_\rho) \|s_k\|, \tag{4.42}$$

then $\|s_k\| = \delta_k^s$ and $\rho_k^f \geq \kappa_\rho$.

Proof. Observe from (4.42) and Lemma 41(i) that $\lambda_k^f > 0$, which along with (4.12c) proves that $\|s_k\| = \delta_k^s$. Next, since $k \in \mathcal{F}$, it must mean that (4.15) is satisfied. It then follows from (4.15b), Lemma 65, (4.12), (4.15d), and (4.15a) that

$$\begin{aligned} m_k^f(0) - m_k^f(s_k) & \geq \kappa_{fm} (m_k^f(n_k) - m_k^f(s_k)) \\ & = \kappa_{fm} (\tfrac{1}{2} t_k^T (H_k + \lambda_k^f I) t_k + \tfrac{1}{2} \lambda_k^f \|t_k\|^2 + \lambda_k^f n_k^T t_k) \\ & \geq \kappa_{fm} (\tfrac{1}{2} - \tfrac{1}{2} \kappa_{ntt}) \lambda_k^f \|t_k\|^2 \geq \tfrac{1}{2} \kappa_{fm} \kappa_{st}^2 (1 - \kappa_{ntt}) \lambda_k^f \|s_k\|^2. \end{aligned} \tag{4.43}$$

Next, the Mean Value Theorem gives the existence of an $\bar{x} \in [x_k, x_k + s_k]$ such that

$f(x_k + s_k) = f_k + g_k^T s_k + \frac{1}{2} s_k^T \nabla^2 f(\bar{x}) s_k$, which with (4.15f) and Assumption 14 gives

$$\begin{aligned}
& m_k^f(s_k) - f(x_k + s_k) \\
&= f_k + g_k^T s_k + \tfrac{1}{2} s_k^T H_k s_k - f_k - g_k^T s_k - \tfrac{1}{2} s_k^T \nabla^2 f(\bar{x}) s_k \\
&= \tfrac{1}{2} s_k^T \left(H_k - \nabla^2 f(x_k) \right) s_k + \tfrac{1}{2} s_k^T \left(\nabla^2 f(x_k) - \nabla^2 f(\bar{x}) \right) s_k \\
&\geq -\tfrac{1}{2} \left\| \left(H_k - \nabla^2 f(x_k) \right) s_k \right\| \|s_k\| - \tfrac{1}{2} \left\| \left(\nabla^2 f(x_k) - \nabla^2 f(\bar{x}) \right) s_k \right\| \|s_k\| \\
&\geq -\tfrac{1}{2} \left(\kappa_{hs} + H_{Lip} \right) \|s_k\|^3.
\end{aligned}$$

Finally, combining the previous inequality, $f_k = m_k^f(0)$, and (4.43), one finds

$$\begin{aligned}
f_k - f(x_k + s_k) &= f_k - m_k^f(s_k) + m_k^f(s_k) - f(x_k + s_k) \\
&\geq \tfrac{1}{2} \kappa_{fm} \kappa_{st}^2 (1 - \kappa_{ntt}) \lambda_k^f \|s_k\|^2 - \tfrac{1}{2} (\kappa_{hs} + H_{Lip}) \|s_k\|^3,
\end{aligned}$$

which combined with (4.42) shows that $\rho_k^f \geq \kappa_\rho$ as desired. □

We now show that a critical ratio increases by at least a constant factor after any unsuccessful F-ITERATION followed by an iteration in which a nonzero tangential step is computed and not reset to zero.

Lemma 67. *If $k \in \mathcal{C}^f$, $\lambda_k^f \geq \underline{\sigma} \|s_k\|$, and $(k+1) \in \mathcal{I}^t$, then $\frac{\lambda_{k+1}^f}{\|s_{k+1}\|} \geq \frac{\lambda_k^f}{\gamma_C \|s_k\|}$.*

Proof. With Lemma 41(i), it follows that $\lambda_k^f \geq \underline{\sigma} \|s_k\| > 0$, meaning that $\|s_k\| = \delta_k^s$. In addition, since $k \in \mathcal{C}^f$, one finds that the condition in Step 12 of Algorithm 9 tests false in iteration k. Hence, Step 16 of Algorithm 9 is reached, meaning, with (4.10), that $\delta_{k+1}^f = \gamma_C \|s_k\| \leq \gamma_C \kappa_\delta \delta_k^v$. Then, from the facts that $\gamma_C < 1$ and $\delta_{k+1}^v \leftarrow \delta_k^v$ (see Step 12 of Algorithm 8), it follows that $\delta_{k+1}^f \leq \kappa_\delta \delta_{k+1}^v$. Consequently, again with (4.10), it follows that $\|s_{k+1}\| = \delta_{k+1}^s = \delta_{k+1}^f = \gamma_C \|s_k\|$. Combining this with the fact that Lemma 47(i) yields $\lambda_{k+1}^f \geq \lambda_k^f$, the result follows. □

Lemma 68. *If $k \in \mathcal{C}^f$ and $(k+1) \in \mathcal{I}^t$, then $\underline{\sigma} \leq \lambda_{k+1}^f / \|s_{k+1}\|$.*

Proof. Since $k \in \mathcal{C}^f$, there are two cases to consider.

Case 1: Step 14 of Algorithm 9 is reached. In this case, it follows that $\|s_{k+1}\| = \delta_{k+1}^f = \|n_k + t(\lambda^f)\|$ with $(t(\lambda^f), \lambda^f)$ computed in Step 13 of Algorithm 9. Together with the fact that $(k+1) \in \mathcal{I}^t$, it follows that $\lambda_{k+1}^f / \|s_{k+1}\| = \lambda^f / \|n_k + t(\lambda^f)\| \geq \underline{\sigma}$.

Case 2: Step 14 of Algorithm 9 is not reached. This only happens if the condition in Step 12 of Algorithm 9 tested false, meaning that $\lambda_k^f/\|s_k\| \geq \underline{\sigma}$. Hence, from Lemma 67, it follows that $\lambda_{k+1}^f/\|s_{k+1}\| \geq \lambda_k^f/\gamma_C\|s_k\|$, which by the facts that $\gamma_C < 1$ and $\lambda_k^f/\|s_k\| \geq \underline{\sigma}$ gives the desired result. □

Next, we provide a bound on the number of iterations in $\mathcal{C}^f$ that may occur before the first or between consecutive iterations in the set $\mathcal{S} \cup \mathcal{V}$.

Lemma 69. *If $\bar{k} \in \mathcal{S} \cup \mathcal{V} \cup \{0\}$, then*

$$|\mathcal{I}_{\mathcal{S}\cup\mathcal{V}}(\bar{k})| \leq 2 + \left\lfloor \frac{1}{\log(\gamma_C^{-1})} \log\left(\frac{\kappa_{hs} + H_{Lip} + 2\kappa_\rho}{\underline{\sigma}\kappa_{fm}\kappa_{st}^2(1-\kappa_{ntt})} \right) \right\rfloor =: K_{\mathcal{C}}^f \geq 0. \tag{4.44}$$

Proof. Let $\bar{k} \in \mathcal{S}\cup\mathcal{V}\cup\{0\}$. Then, $\mathcal{I}_{\mathcal{S}\cup\mathcal{V}}(\bar{k}) \subseteq \mathcal{C}^f$. The result follows trivially if $|\mathcal{I}_{\mathcal{S}\cup\mathcal{V}}(\bar{k})| \leq 1$. Therefore, for the remainder of the proof, let us assume that $|\mathcal{I}_{\mathcal{S}\cup\mathcal{V}}(\bar{k})| \geq 2$. It follows from Lemma 68, $\bar{k}+1 \in \mathcal{C}^f$, and $\bar{k}+2 \in \mathcal{C}^f \subseteq \mathcal{F}$ (meaning that $t_{\bar{k}+2} \neq 0$ and $(\bar{k}+2) \in \mathcal{I}^t$) that $\underline{\sigma} \leq \lambda_{\bar{k}+2}^f/\|s_{\bar{k}+2}\|$. Combining this inequality with Lemma 66, Lemma 67, the fact that and $(k_{\mathcal{S}\cup\mathcal{V}}(\bar{k})-1) \in \mathcal{C}^f$ to get

$$\underline{\sigma}\left(\frac{1}{\gamma_C}\right)^{(k_{\mathcal{S}\cup\mathcal{V}}(\bar{k})-1)-(\bar{k}+2)} \leq \frac{\lambda^f_{k_{\mathcal{S}\cup\mathcal{V}}(\bar{k})-1}}{\|s_{k_{\mathcal{S}\cup\mathcal{V}}(\bar{k})-1}\|} \leq \left(\frac{\kappa_{hs} + H_{Lip} + 2\kappa_\rho}{\kappa_{fm}\kappa_{st}^2(1-\kappa_{ntt})} \right).$$

The desired result now follows since $|\mathcal{I}_{\mathcal{S}\cup\mathcal{V}}(\bar{k})| = k_{\mathcal{S}\cup\mathcal{V}}(\bar{k}) - \bar{k} - 1$. □

We have now arrived at our complexity result for phase 1.

Theorem 13. *For a scalar $\epsilon \in (0,\infty)$, the cardinality of $\mathcal{I}$ is at most*

$$K(\epsilon) := 1 + (K_\sigma(\epsilon) + K_\Delta^v)(K_{\mathcal{C}}^v + 1)K_{\mathcal{C}}^f, \tag{4.45}$$

where $K_\sigma(\epsilon)$, K_Δ^v, $K_{\mathcal{C}}^v$, and $K_{\mathcal{C}}^f$ are defined in Lemmas 61, 62, 64, and 69, respectively. Consequently, for any $\bar{\epsilon} \in (0,\infty)$, it follows that $K(\epsilon) = \mathcal{O}(\epsilon^{-3/2})$ for all $\epsilon \in (0,\bar{\epsilon})$.

Proof. Without loss of generality, let us assume that at least one iteration is performed. Then, Lemmas 61 and 62 guarantee that at most $K_\sigma(\epsilon) + K_\Delta^v$ successful iterations are included in $\mathcal{I}$. In addition, Lemmas 49, 64, and 69 guarantee that, before each successful iteration, there can be at most $(K_{\mathcal{C}}^v + 1)K_{\mathcal{C}}^f$ unsuccessful iterations. Also accounting for the first iteration, the desired result follows. □

Building on this theorem, one may now apply the analysis in [19, §3.1–3.2] to arrive at the following corollary involving practical termination conditions for phase 1.

Corollary 1. *Let $(\epsilon_{feas}, \epsilon_{inf}) \in (0,1) \times (0,1)$ be given constants. Then, the number of iterations required until either*

$$\|c_k\| \leq \epsilon_{feas} \tag{4.46a}$$

$$\text{or} \quad \|J_k^T c_k\| \leq \epsilon_{inf} \|c_k\| \tag{4.46b}$$

is satisfied, is at most $\mathcal{O}(\max\{\epsilon_{feas}^{-1/2}, \epsilon_{inf}^{-3/2}\})$.

Proof. The result follows in the same manner as [19, Th. 3.2]. In particular, the only property of the phase 1 algorithm needed for the proof in [19] is that reductions in v for successful steps are at least a fraction of $\|g_k^v\|^{3/2}$. The same holds for our algorithm with respect to $\{v_k^{\max}\}$, as shown by Lemmas 43 and 60. □

If the constraint Jacobians encountered by the algorithm are not rank deficient (and do not tend toward rank deficiency), then the following corollary gives a similar result as that above, but for an infeasibility measure. This occurs, e.g., if all iterates and all limit points of the algorithm are points at which the linear independence constraint qualification (LICQ) is satisfied.

Corollary 2. *Suppose that, for all $k \in \mathbb{N}$, the constraint Jacobian J_k has full row rank with singular values bounded below by $\zeta_{\min} \in (0, \infty)$. Then, for $\epsilon \in (0, \infty)$, the cardinality of $\mathcal{I}_c := \{k \in \mathbb{N} : \|c_k\| > \epsilon/\zeta_{\min}\}$, is at most $K(\epsilon)$ defined in (4.45). Consequently, for any $\bar{\epsilon} \in (0, \infty)$, the cardinality of $\mathcal{I}_c$ is $\mathcal{O}(\epsilon^{-3/2})$ for all $\epsilon \in (0, \bar{\epsilon})$.*

Proof. Under the stated conditions, $\|g_k^v\| \equiv \|J_k^T c_k\| \geq \zeta_{\min} \|c_k\|$ for all $k \in \mathcal{I}$. Thus, since $\|g_k^v\| \leq \epsilon$ implies $\|c_k\| \leq \epsilon/\zeta_{\min}$, the result follows from Theorem 13. □

4.4 Phase 2: Obtaining Optimality

A complete algorithm for solving problem (4.1) proceeds as follows. The phase 1 method, Algorithm 8, is run until either an approximate feasible point or approximate infeasible stationary point is found, i.e., for some $(\epsilon_{feas}, \epsilon_{inf}) \in (0,1) \times (0,1)$, the method is run until (4.46) holds for some $k \in \mathbb{N}$. If phase 1 terminates with (4.46a) failing to hold and

(4.46b) holding, then the entire algorithm is terminated with a declaration of having found an infeasible (approximately) stationary point. Otherwise, if (4.46a) holds, then a phase 2 method is run that maintains at least ϵ_{feas}-feasibility while seeking optimality.

There are various options for phase 2. For example, respecting the current state-of-the-art nonlinear optimization methods, one can run a trust funnel method such as that in [38]. One can even run such a method with the initial trust funnel radius for $v(x) = \frac{1}{2}\|c(x)\|^2$ set at $\frac{1}{2}\epsilon_{feas}^2$ so that ϵ_{feas}-feasibility will be maintained as optimality is sought. We do not claim worst-case iteration complexity guarantees for such a method, though empirical evidence suggests that it would perform well. If c is affine, then one could run a method, such as the ARC method from [14, 15] (see also the previous work in [45, 56, 61]) or the TRACE method from Chapter 2, where steps toward reducing the objective are restricted to the null space of the constraint Jacobian. For such a reduced-space method, ϵ_{feas}-feasibility will be maintained while the analyses in [14, 15] and Chapter 2 guarantee that the number of iterations required to reduce the norm of the reduced gradient below a given tolerance $\epsilon_{opt} \in (0, \infty)$ is at most $\mathcal{O}(\epsilon_{opt}^{-3/2})$. With $\epsilon = \epsilon_{opt} = \epsilon_{feas}$, this gives an overall (phase 1 + phase 2) complexity of $\mathcal{O}(\epsilon^{-3/2})$.

More interesting for our purposes are phase 2 approaches designed with an eye toward attaining good complexity properties. To achieve this, one can run the objective-target-following approach stated as [19, Alg. 4.1, Phase 2] or as [6, Alg. 2.1, Phase 2]. These approaches apply an unconstrained optimization algorithm to minimize the residual $\Phi : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}$ defined by $\Phi(x, t) = \frac{1}{2}\|r(x, t)\|^2$ where $r : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}$ has

$$r(x, t) = (c(x), f(x) - t). \tag{4.47}$$

Updated dynamically by the algorithm, the parameter t may be viewed as a target value for reducing the objective function value.

Our goal here is to show that a phase 2 method can be built upon the TRACE method from Chapter 2 yielding the same worst-case complexity properties as the ARC-based method in [19]. For our algorithm and analysis of it, we assume the following.

Assumption 15. *The initial objective value for phase 2, namely, f_0, is bounded above independently from ϵ_{feas}. For all $x \in \mathbb{R}^N$ with $\|c(x)\| \leq \epsilon_{feas} \in (0, \infty)$, the objective f is bounded below by $f_{\min} \in \mathbb{R}$. The functions f and c and their first and second derivatives are Lipschitz continuous on the path defined by all phase 2 iterates.*

Our phase 2 algorithm is stated as Algorithm 11. We refer the reader to [19] for further details on the design of the algorithm and to Chapter 2 for further details on TRACE. In short, in iteration $k \in \mathbb{N}$, the subsequent iterate x_{k+1} is computed to reduce $\Phi(\cdot, t_k)$ while the subsequent target t_{k+1} is chosen to ensure the relationships in the following lemma (whose proof follows that of [19, Lemma 4.1]).

Algorithm 11 TRACE Algorithm for Phase 2

Require: termination tolerance $\epsilon \in (0, \infty)$ and $x_0 \in \mathbb{R}^N$ with $\|c_0\| \leq \epsilon_{feas} \in (0, 1)$

1: **procedure** TRACE_PHASE_2
2: set $t_0 \leftarrow f_0 - \sqrt{\epsilon_{feas}^2 - \|c_0\|^2}$
3: **for** $k \in \mathbb{N}$ **do**
4: perform one iteration of TRACE toward minimizing $\Phi(x, t_k)$ to compute s_k
5: **if** s_k is an acceptable step **then**
6: set $x_{k+1} \leftarrow x_k + s_k$ (and other quantities following TRACE)
7: **if** $r(x_{k+1}, t_k) \neq 0$ and $\|\nabla_x \Phi(x_{k+1}, t_k)\| \leq \epsilon \|r(x_{k+1}, t_k)\|$ **then**
8: terminate
9: **else**
10: set $t_{k+1} \leftarrow f(x_{k+1}) - \sqrt{\|r(x_k, t_k)\|^2 - \|r(x_{k+1}, t_k)\|^2 + \left(f(x_{k+1}) - t_k\right)^2}$
11: **else**
12: set $x_{k+1} \leftarrow x_k$ (and other quantities following TRACE)
13: set $t_{k+1} \leftarrow t_k$

Lemma 70. *Let $\epsilon_{feas} \in (0, 1)$ be given. For all $k \in \mathbb{N}$, it follows that*

$$t_{k+1} \leq t_k, \tag{4.48a}$$

$$0 \leq f(x_k) - t_k \leq \epsilon_{feas}, \tag{4.48b}$$

$$\|r(x_k, t_k)\| = \epsilon_{feas}, \tag{4.48c}$$

$$\text{and} \quad \|c(x_k)\| \leq \epsilon_{feas}. \tag{4.48d}$$

Proof. Note that, in TRACE, the objective function is monotonically nonincreasing; see (2.5) and Step 5 in Algorithm 6. Hence, each acceptable step s_k computed in Algorithm 11 yields $\Phi(x_{k+1}, t_k) \leq \Phi(x_k, t_k)$, from which it follows that the value for t_{k+1} in Step 10 is well-defined. Then, since all definitions and procedures in Algorithm 11 that yield (4.48) are exactly the same as in [19, Alg. 4.1], a proof for the inequalities in (4.48) is given by the proof of [19, Lemma 4.1]. □

In the next lemma, we recall a critical result from Chapter 2, arguing that it remains true for Algorithm 11 under our assumptions about the problem functions.

Lemma 71. *Let $\{\sigma_k\}$ be generated as in* TRACE. *(See Algorithm 6.) Then, there exists a scalar constant $\sigma_{\max} \in (0, \infty)$ such that $\sigma_k \leq \sigma_{\max}$ for all $k \in \mathbb{N}$.*

Proof. The result follows similarly as Lemma 15. Here, similar to [19, §5], it is important to note that Assumption 15 ensures that Φ and its first and second derivatives are globally Lipschitz continuous on a path defined by the phase 2 iterates. This ensures that results of the kind given as Lemmas 13 and 14 hold true, which are necessary for proving Lemma 15 (for bounding $\{\sigma_k\}$ above by a $\sigma_{\max} \in (0, \infty)$). □

We now argue that the number of iterations taken for any fixed value of the target for the objective function is bounded above by a positive constant.

Lemma 72. *The number of iterations required before the first accepted step or between two successive accepted steps with a fixed target is bounded above by*

$$K_t := 2 + \left\lfloor \frac{1}{\log(\min\{\gamma_\lambda, \gamma_C^{-1}\})} \log\left(\frac{\sigma_{\max}}{\underline{\sigma}}\right) \right\rfloor,$$

where the constants $\gamma_\lambda \in (0, \infty)$, $\gamma_C \in (0, 1)$, are $\underline{\sigma} \in (0, \infty)$ are parameters used by TRACE *(see Algorithm 6) that are independent of k and satisfy $\underline{\sigma} \leq \sigma_{\max}$.*

Proof. The properties of TRACE corresponding to so-called *contraction* and *expansion* iterations all hold for Algorithm 11 for sequences of iterations in which a target value is held fixed. Therefore, the result follows by Lemmas 19 and 21, which combined show that the maximum number of iterations of interest is equal to the maximum number of contractions that may occur plus one. □

The next lemma merely states a fundamental property of TRACE.

Lemma 73. *Let $H_\Phi \in (0, \infty)$ be the Lipschitz constant for the Hessian function of Φ along the path of phase 2 iterates and let $\eta \in (0, 1)$ be the acceptance constant from* TRACE. *Then, for x_{k+1} following an accepted step s_k, it follows that*

$$\Phi(x_k, t_k) - \Phi(x_{k+1}, t_k) \geq \eta(H_\Phi + \sigma_{\max})^{-3/2} \|\nabla_x \Phi(x_{k+1}, t_k)\|^{3/2}.$$

Proof. With Lemma 71 and adapting the conclusion of Lemma 16, the result follows as in the beginning of the proof of Lemma 17. □

The preceding lemma allows us to prove the following useful result.

Lemma 74. *For x_{k+1} following an accepted step s_k yielding*

$$\|\nabla_x \Phi(x_{k+1}, t_k)\| > \epsilon \|r(x_{k+1}, t_k)\|, \tag{4.49}$$

it follows that $\|r(x_k, t_k)\| - \|r(x_{k+1}, t_k)\| \geq \kappa_t \min\{\epsilon^{3/2}\epsilon_{feas}^{1/2}, \epsilon_{feas}\}$, where $\beta \in (0,1)$ is any fixed problem-independent constant, $\omega := \eta(H_\Phi + \sigma_{\max})^{-3/2} \in (0, \infty)$ is the constant appearing in Lemma 73, and $\kappa_t := \min\{\omega\beta^{3/2}, 1-\beta\}$.

Proof. Along with the result of Lemma 73, it follows that

$$\begin{aligned}
\|r(x_k, t_k)\|^2 - \|r(x_{k+1}, t_k)\|^2 &\geq 2\omega \|\nabla_x \Phi(x_{k+1}, t_k)\|^{3/2} \\
&= 2\omega \left(\frac{\|\nabla_x \Phi(x_{k+1}, t_k)\|}{\|r(x_{k+1}, t_k)\|} \right)^{3/2} \|r(x_{k+1}, t_k)\|^{3/2} \\
&\geq 2\omega\epsilon^{3/2} \|r(x_{k+1}, t_k)\|^{3/2}.
\end{aligned}$$

Then, if $\|r(x_{k+1}, t_k)\| > \beta \|r(x_k, t_k)\|$, it follows with (4.48c) that

$$\|r(x_k, t_k)\|^2 - \|r(x_{k+1}, t_k)\|^2 \geq 2\omega\epsilon^{3/2}\beta^{3/2} \|r(x_k, t_k)\|^{3/2} = 2\omega\epsilon^{3/2}\beta^{3/2}\epsilon_{feas}^{3/2}, \tag{4.50}$$

from which it follows along with $\|r(x_{k+1}, t_k)\| \leq \|r(x_k, t_k)\|$ that

$$\begin{aligned}
\|r(x_k, t_k)\| - \|r(x_{k+1}, t_k)\| &= \frac{\|r(x_k, t_k)\|^2 - \|r(x_{k+1}, t_k)\|^2}{\|r(x_k, t_k)\| + \|r(x_{k+1}, t_k)\|} \\
&\geq \frac{\|r(x_k, t_k)\|^2 - \|r(x_{k+1}, t_k)\|^2}{2\|r(x_k, t_k)\|} \\
&= \frac{\|r(x_k, t_k)\|^2 - \|r(x_{k+1}, t_k)\|^2}{2\epsilon_{feas}} \geq \omega\epsilon^{3/2}\beta^{3/2}\epsilon_{feas}^{1/2}.
\end{aligned}$$

On the other hand, if $\|r(x_{k+1}, t_k)\| \leq \beta \|r(x_k, t_k)\|$, then using (4.48c) it follows that

$$\|r(x_k, t_k)\| - \|r(x_{k+1}, t_k)\| \geq (1-\beta)\|r(x_k, t_k)\| = (1-\beta)\epsilon_{feas}.$$

Combining the results of both cases, the desired conclusion follows. □

We next prove a lower bound on the decrease of the target value.

Lemma 75. *Suppose that the termination tolerance is set so that* $\epsilon \leq \epsilon_{feas}^{1/3}$. *Then, for* x_{k+1} *following an accepted step* s_k *such that the termination conditions in Step 7 are not satisfied, it follows that, with* $\kappa_t \in (0,1)$ *defined as in Lemma 74,*

$$t_k - t_{k+1} \geq \kappa_t \epsilon^{3/2} \epsilon_{feas}^{1/2}. \tag{4.51}$$

Proof. A proof follows similarly to that of [19, Lem. 5.3]. In particular, if the reason that the termination conditions in Step 7 are not satisfied is because (4.49) holds, then Lemma 74 and the fact that $\epsilon \leq \epsilon_{feas}^{1/3}$ imply that

$$\|r(x_k, t_k)\| - \|r(x_{k+1}, t_k)\| \geq \kappa_t \min\{\epsilon^{3/2}\epsilon_{feas}^{1/2}, \epsilon_{feas}\} \geq \kappa_t \epsilon^{3/2} \epsilon_{feas}^{1/2}.$$

On the other hand, if the condition in Step 7 is not satisfied because $\|r(x_{k+1}, t_k)\| = 0$, it follows from (4.48c), $\kappa_t \in (0,1)$, and $\epsilon \leq \epsilon_{feas}^{1/3}$ that $\|r(x_k, t_k)\| - \|r(x_{k+1}, t_k)\| = \epsilon_{feas} \geq \kappa_t \epsilon^{3/2} \epsilon_{feas}^{1/2}$. Combining these two cases, (4.47), and (4.48c), one finds that

$$(f(x_{k+1}) - t_k)^2 + \|c(x_{k+1})\|^2 = \|r(x_{k+1}, t_k)\|^2 \leq (\epsilon_{feas} - \kappa_t \epsilon^{3/2} \epsilon_{feas}^{1/2})^2.$$

Now, from Step 10 of Algorithm 11, (4.47), and (4.48c), it follows that

$$\begin{aligned} t_k - t_{k+1} &= -(f(x_{k+1}) - t_k) + \sqrt{\|r(x_k, t_k)\|^2 - \|r(x_{k+1}, t_k)\|^2 + (f(x_{k+1}) - t_k)^2} \\ &= -(f(x_{k+1}) - t_k) + \sqrt{\|r(x_k, t_k)\|^2 - \|c(x_{k+1})\|^2} \\ &= -(f(x_{k+1}) - t_k) + \sqrt{\epsilon_{feas}^2 - \|c(x_{k+1})\|^2}. \end{aligned}$$

Overall, it follows that [19, Lemma 5.2] can be applied (with "f" = $f(x_{k+1}) - t_k$, "c" = $\|c(x_{k+1})\|$, "ϵ" = ϵ_{feas}, and "τ" = $\epsilon_{feas} - \kappa_t \epsilon^{3/2} \epsilon_{feas}^{1/2}$) to obtain the result. □

We now show that if the termination condition in Step 7 of Algorithm 11 is never satisfied, then the algorithm takes infinitely many accepted steps.

Lemma 76. *If Algorithm 11 does not terminate finitely, then it takes infinitely many accepted steps.*

Proof. To derive a contradiction, suppose that the number of accepted steps is finite. Then, since it does not terminate finitely, there exists $\bar{k} \in \mathbb{N}$ such that s_k is not acceptable

for all $k \geq \bar{k}$. Therefore, by the construction of the algorithm, it follows that $t_k = t_{\bar{k}}$ for all $k \geq \bar{k}$. This means that the algorithm proceeds as if the TRACE algorithm (Alg. 6) is being employed to minimize $\Phi(\cdot, t_{\bar{k}})$, from which it follows by Lemmas 6, 7, and 8 that, for some sufficiently large $k \geq \bar{k}$, an acceptable step s_k will be computed. This contradiction completes the proof. □

Before proceeding, let us discuss the situation in which the termination conditions in Step 7 of Algorithm 11 are satisfied. This discussion, originally presented in [19], justifies the use of these termination conditions.

Suppose $\|r(x_{k+1}, t_k)\| \neq 0$ and $\|\nabla_x \Phi(x_{k+1}, t_k)\| \leq \epsilon \|r(x_{k+1}, t_k)\|$. If $f_{k+1} = t_k$, then these mean that $\|c_{k+1}\| \neq 0$ and $\|\nabla_x \Phi(x_{k+1}, t_k)\| \leq \epsilon \|c_{k+1}\|$, which along with $\nabla_x \Phi(x_{k+1}, t_k) = J_{k+1}^T c_{k+1} + (f_{k+1} - t_k) g_{k+1}$ would imply that $\|J_{k+1}^T c_{k+1}\| \leq \epsilon \|c_{k+1}\|$. That is, under these conditions, x_{k+1} is an approximate first-order stationary point for minimizing $\|c\|$. If $f_{k+1} \neq t_k$, then the satisfied termination conditions imply that $\|J_{k+1}^T c_{k+1} + (f_{k+1} - t_k) g_{k+1}\| / \|r(x_{k+1}, t_k)\| \leq \epsilon$. By dividing the numerator and denominator of the left-hand side by $f(x_{k+1}) - t_k > 0$ (recall (4.48b)), defining

$$y(x_{k+1}, t_k) := c(x_{k+1}) / (f(x_{k+1}) - t_k) \in \mathbb{R}^M, \tag{4.52}$$

and substituting $y(x_{k+1}, t_k)$ back into the inequality, one finds that

$$\|J_{k+1}^T y(x_{k+1}, t_k) + g_{k+1}\| / \|(y(x_{k+1}, t_k), 1)\| \leq \epsilon. \tag{4.53}$$

As argued in [19], one may use a perturbation argument to say that $(x_{k+1}, y(x_{k+1}, t_k))$ satisfying the *relative KKT error* conditions (4.53) and $\|c_{k+1}\| \leq \epsilon_{feas}$ corresponds to a first-order stationary point for problem (4.1). Specifically, consider $x = x_* + \delta_x$ and $y = y_* + \delta_y$ where (x_*, y_*) is a primal-dual pair satisfying the KKT conditions for problem (4.1). Then, a first-order Taylor expansion of $J(x_*)^T y_* + g(x_*)$ to estimate its value at (x, y) yields the estimate $(\nabla^2 f(x_*) + \sum_{i=1}^M [y_i]_* \nabla^2 c_i(x^*)) \delta_x + J(x^*)^T \delta_y$. The presence of the dual variable y^* in this estimate confirms that the magnitude of the dual variable should not be ignored in a relative KKT error condition such as (4.53).

We now prove our worst-case iteration complexity result for phase 2.

Theorem 14. *Suppose that the termination tolerances are set so that* $\epsilon \leq \epsilon_{feas}^{1/3}$ *with* $\epsilon_{feas} \in (0, 1)$. *Then, Algorithm 11 requires at most* $\mathcal{O}(\epsilon^{-3/2} \epsilon_{feas}^{-1/2})$ *iterations until the*

termination condition in Step 7 is satisfied, at which point either

$$\|J_{k+1}^T y(x_{k+1}, t_k) + g_{k+1}\| / \|(y(x_{k+1}, t_k), 1)\| \leq \epsilon \text{ and } \|c_{k+1}\| \leq \epsilon_{feas} \tag{4.54a}$$

$$\text{or } \|J_{k+1}^T c_{k+1}\| / \|c_{k+1}\| \leq \epsilon \text{ and } \|c_{k+1}\| \leq \epsilon_{feas} \tag{4.54b}$$

is satisfied, with $y(x_{k+1}, t_k)$ defined in (4.52).

Proof. Recall that if the termination condition in Step 7 is satisfied for some $k \in \mathbb{N}$, then, by the arguments prior to the lemma, either (4.54a) or (4.54b) will be satisfied. Thus, we aim to show an upper bound on the number of iterations required by the algorithm until the termination condition in Step 7 is satisfied.

Without loss of generality, let us suppose that the algorithm performs at least one iteration. Then, we claim that there exists some $k \in \mathbb{N}$ such that the termination condition does not hold for (x_k, t_{k-1}), but does hold for (x_{k+1}, t_k). To see this, suppose for contradiction that the termination condition is never satisfied. Then, by Lemma 75, it follows that for all $k \in \mathbb{N}$ such that s_k is acceptable one finds that (4.51) holds. This, along with Lemma 76, implies that $\{t_k\} \searrow -\infty$. However, this along with (4.48b) implies that $\{f_k\} \searrow -\infty$, which contradicts Assumption 15.

Now, since the termination condition is satisfied at (x_{k+1}, t_k), but not in the iteration before, it follows that s_k must be an acceptable step. Hence, from Assumption 15, (4.48b), Lemma 75, Step 2, it follows that

$$f_{\min} \leq f_k \leq t_k + \epsilon_{feas} \leq t_0 - K_{\mathcal{A}} \kappa_t \epsilon^{3/2} \epsilon_{feas}^{1/2} + \epsilon_{feas} \leq f(x_0) - K_{\mathcal{A}} \kappa_t \epsilon^{3/2} \epsilon_{feas}^{1/2} + \epsilon_{feas},$$

where $K_{\mathcal{A}}$ is the number of accepted steps prior to iteration $(k+1)$. Rearranging and since $\epsilon_{feas} \in (0, 1)$, one finds that $K_{\mathcal{A}} \leq \left\lceil (f_0 + \epsilon_{feas} - f_{\min} + 1) / (\kappa_t \epsilon^{3/2} \epsilon_{feas}^{1/2}) \right\rceil$. From this and Lemma 72, the desired result follows. □

This should be viewed in two ways. First, if $\epsilon = \epsilon_{feas}^{2/3}$, then the overall complexity is $\mathcal{O}(\epsilon_{feas}^{-3/2})$, though of course this corresponds to a looser tolerance on the relative KKT error than on feasibility. Second, if $\epsilon = \epsilon_{feas}$ (so that the two tolerances are equal), then the overall complexity is $\mathcal{O}(\epsilon_{feas}^{-2})$.

4.5 Numerical Experiments

Our goal now is to demonstrate that instead of having a phase 1 method that solely seeks (approximate) feasibility (such as in [6, 19]), it is beneficial to employ a phase 1 method that also attempts to reduce the objective. To show this, a Matlab implementation of our phase 1 (Algorithm 8) plus a phase 2 has been written. The implementation has two modes: one following Algorithm 8 and one employing the same procedures except that the tangential step t_k is set to zero for all $k \in \mathbb{N}$ so that all iterations are V-ITERATIONs. We refer to the former implementation as TF and the latter as TF-V-ONLY. For phase 2 for both methods, we implemented a trust funnel method based on that proposed in [38] with the modification that the normal step computation is never skipped. The initial value of $v_k^{\max}$ for phase 2 is the final value obtained from phase 1; note that this means that phase 2 does not necessarily maintain (near) feasibility as would Algorithm 11. In both phases 1 and 2, subproblems are solved to high accuracy using a Matlab implementation of the trust region subproblem solver described as [23, Alg. 7.3.4], which goes back to the work in [52]. However, if a step is rejected and a contraction is performed, then, when possible, rather than solve a trust region subproblem, we compute steps by solving linear systems (to solve $\mathcal{Q}_k^v(\cdot)$ and/or $\mathcal{Q}_k^f(\cdot)$ for a given dual variable). The fact that the normal step computation is never skipped and the subproblems are always solved to high accuracy allows our implementation to ignore so-called "y-iterations" [38].

Phase 1 in each implementation terminates in iteration $k \in \mathbb{N}$ if either

$$\|c_k\|_\infty \leq 10^{-6} \max\{\|c_0\|_\infty, 1\} \quad \text{or} \quad \begin{cases} \|J_k^T c_k\|_\infty \leq 10^{-9} \max\{\|J_0^T c_0\|_\infty, 1\} \\ \text{and } \|c_k\|_\infty > 10^{-3} \max\{\|c_0\|_\infty, 1\}. \end{cases} \tag{4.55}$$

Phase 2 terminates in iteration $k \in \mathbb{N}$ if the second condition in (4.55) holds or if both the first condition in (4.55) holds and, with y_k computed as least squares multipliers for all $k \in \mathbb{N}$, one finds $\|g_k + J_k^T y_k\|_\infty \leq 10^{-9} \max\{\|g_0 + J_0^T y_0\|_\infty, 1\}$. Input parameters used in the code are stated in Table 4.1. The only values that do not appear are κ_ρ and γ_C. For κ_ρ, for simplicity we employed this in both (4.15c) and (4.17), as well as in the step acceptance conditions in Step 2 in Algorithm 9 and Step 2 in Algorithm 10. However, it turns out that our analysis can be adapted to handle different values in these places. Along these lines, our code uses $\kappa_\rho = 10^{-12}$ in (4.15c) and (4.17), but uses $\kappa_\rho = 10^{-8}$

in the step acceptance conditions. For γ_C, our code uses 0.5 in Algorithm 9 and 10^{-2} in Algorithm 10, where again our analysis can be extended to allow using different constants in these places. We chose these constant values since they worked well for both algorithms in our tests. For all $k \in \mathbb{N}$, we set H_k as the Hessian of the Lagrangian, which is set using least squares multipliers.

Table 4.1: Input parameters for TF and TF-V-ONLY.

κ_n	9e-01	κ_{st}	1e-12	κ_p	1e-06	κ_δ	1e+02
κ_{vm}	1e-12	κ_{ntt}	1-(2e-12)	κ_{ht}	1e+20	γ_E	2e+00
κ_{ntn}	1e-12	κ_{v1}	9e-01	κ_{hs}	1e+20	γ_λ	2e+00
κ_{fm}	1e-12	κ_{v2}	9e-01	$\underline{\sigma}$	1e-12	$\overline{\sigma}$	1e+20

We ran TF and TF-V-ONLY to solve the equality constrained problems in the `CUTEst` test set [43]. In particular, we ran our experiments on a machine with 8GB of memory and set a time limit of four hours. Among 190 such problems, we removed 78 that had a constant (or null) objective, 17 for which phase 1 of both algorithms terminated immediately at the initial point due to the first condition in (4.55), one for which phase 1 of both algorithms terminated immediately at the initial point due to the second condition in (4.55), 26 for which both algorithms had insufficient memory, 19 on which both algorithms exceeded our time limit, and 8 problems for which both algorithms failed (due to different combinations of the reasons above, namely, memory limit, time limit, etc.). In addition, one problem was removed because TF found a relative stationary point but TF-V-ONLY failed due to a subproblem solver error, and two were removed because TF-V-ONLY found relative stationary points but TF failed because of our memory limit. The remaining set consisted of 38 problems.

The results we obtained are provided in Table 4.2. For each problem, we indicate the number of variables (n), number of equality constraints (m), number of V-ITERATIONS (`#V`), number of F-ITERATIONS (`#F`), objective function value at the end of phase 1 (f), and dual infeasibility value at the end of phase 1 ($\|g + J^T y\|$). We write `INF` to indicate that an algorithm did not run phase 2 due to the fact that phase 1 ended with an infeasible stationary point (i.e., the second condition in (4.55) was satisfied). The results illustrate that, within a comparable number of iterations, TF typically yields better final points from phase 1. This can be seen in the fact that the objective at the end of

phase 1, dual infeasibility at the end of phase 1, and the number of iterations required in phase 2 are all typically smaller for TF than they are for TF-V-ONLY. Note that for some problems, such as BT1, TF only performs V-ITERATIONs in phase 1, yet yields a better final point than does TF-V-ONLY; this occurs since the phase 1 iterations in TF may involve nonzero tangential steps.

Table 4.2: Numerical results for TF and TF-V-ONLY.

			TF						TF-V-ONLY				
			Phase 1				Phase 2		Phase 1			Phase 2	
Problem	n	m	#V	#F	f	$\|g + J^T y\|$	#V	#F	#V	f	$\|g + J^T y\|$	#V	#F
BT1	2	1	4	0	-8.02e-01	+4.79e-01	0	139	4	-8.00e-01	+7.04e-01	7	137
BT10	2	2	10	0	-1.00e+00	+5.39e-04	1	0	10	-1.00e+00	+6.74e-05	1	0
BT11	5	3	6	1	+8.25e-01	+4.84e-03	2	0	1	+4.55e+04	+2.57e+04	16	36
BT12	5	3	46	6	+6.57e+00	+2.06e-01	6	11	16	+3.34e+01	+4.15e+00	4	8
BT2	3	1	22	8	+1.45e+03	+3.30e+02	3	12	20	+6.14e+04	+1.82e+04	0	40
BT3	5	3	1	0	+4.09e+00	+6.43e+02	1	0	1	+1.01e+05	+8.89e+02	0	1
BT4	3	2	1	0	-1.86e+01	+9.77e+00	20	11	1	-1.86e+01	+9.77e+00	20	11
BT5	3	2	13	6	+9.67e+02	+4.75e+00	4	74	8	+9.62e+02	+7.38e-01	5	1
BT6	5	2	11	45	+2.77e-01	+4.64e-02	1	0	14	+5.81e+02	+4.50e+02	5	59
BT7	5	3	15	6	+1.31e+01	+5.57e+00	5	1	12	+1.81e+01	+1.02e+01	19	28
BT8	5	2	22	2	+1.00e+00	+4.29e-04	0	1	10	+2.00e+00	+2.00e+00	1	97
BT9	4	2	11	1	-1.00e+00	+8.56e-05	1	0	10	-9.69e-01	+2.26e-01	5	1
BYRDSPHR	3	2	29	2	-4.68e+00	+1.28e-05	0	0	19	-5.00e-01	+1.00e+00	16	5
CHAIN	800	401	9	0	+5.12e+00	+2.35e-04	3	20	9	+5.12e+00	+2.35e-04	3	20
FLT	2	2	15	4	+2.68e+10	+3.28e+05	0	13	19	+2.68e+10	+3.28e+05	0	17
GENHS28	10	8	1	0	+9.27e-01	+5.83e+01	0	0	1	+2.49e+03	+1.11e+02	0	1
HS100LNP	7	2	16	2	+6.89e+02	+1.74e+01	3	39	5	+7.17e+02	+1.97e+01	13	51
HS111LNP	10	3	9	1	-4.78e+01	+4.91e-06	2	0	10	-4.62e+01	+7.49e-01	10	1
HS27	3	1	2	0	+8.77e+01	+2.03e+02	3	5	1	+2.54e+01	+1.41e+02	11	34
GENHS28	10	8	1	0	+9.27e-01	+5.83e+01	0	0	1	+2.49e+03	+1.11e+02	0	1
HS39	4	2	11	1	-1.00e+00	+8.56e-05	1	0	10	-9.69e-01	+2.26e-01	5	1
HS40	4	3	4	0	-2.50e-01	+1.95e-06	0	0	3	-2.49e-01	+3.35e-02	2	1
HS42	4	2	4	1	+1.39e+01	+3.94e-04	1	0	1	+1.50e+01	+2.00e+00	3	1
HS52	5	3	1	0	+5.33e+00	+1.90e+02	1	0	1	+8.65e+03	+3.86e+02	0	1
HS6	2	1	1	0	+4.84e+00	+1.56e+00	32	136	1	+4.84e+00	+1.56e+00	32	136
HS61	3	2	1	0	-5.88e+01	+2.40e+01	INF	INF	1	-5.88e+01	+2.40e+01	INF	INF
HS7	2	1	7	1	-2.35e-01	+1.18e+00	7	2	8	+3.79e-01	+1.07e+00	5	2
HS77	5	2	13	30	+2.42e-01	+1.26e-02	0	0	17	+5.52e+02	+4.54e+02	3	11
HS78	5	3	6	0	-2.92e+00	+3.65e-04	1	0	10	-1.79e+00	+1.77e+00	2	30
HS79	5	3	13	21	+7.88e-02	+5.51e-02	0	2	10	+9.70e+01	+1.21e+02	0	24
LINCONT	249	419	1	0	+0.00e+00	+0.00e+00	INF	INF	1	+0.00e+00	+0.00e+00	INF	INF
LUKVLE1	10000	9998	7	5	+7.87e-01	+4.18e-04	0	0	19	+1.09e+09	+4.53e+07	0	238
LUKVLE3	10000	2	9	7	+2.50e+00	+8.24e-01	0	5	12	+8.68e+06	+5.85e+05	0	79
MARATOS	2	1	4	0	-1.00e+00	+8.59e-05	1	0	3	-9.96e-01	+9.02e-02	2	1
MWRIGHT	5	3	17	6	+2.31e+01	+5.78e-05	1	0	7	+5.07e+01	+1.04e+01	12	20
OPTCTRL3	4499	3000	1	9	+3.51e+01	+1.83e+01	0	15	4	+1.15e+05	+4.57e+03	1	21
OPTCTRL6	4499	3000	1	9	+3.51e+01	+1.83e+01	0	15	4	+1.15e+05	+4.57e+03	1	21
ORTHREGB	27	6	10	13	+1.79e-04	+2.71e+06	0	31	10	+2.73e+00	+1.60e+00	0	10

Chapter 5

Conclusion and Future Work

In this dissertation, we tried to fill the gap between the theoretical guarantees of trust region based methods and their superior practical performance. Through carefully designed modifications within a trust region framework, we showed that one of the most popular and effective algorithms in the nonlinear optimization literature can also achieve ideal theoretical behavior as well, namely, optimal iteration complexity guarantees. By proposing an intentionally general inexact regularized Newton framework, we suggested that a carefully generalized Newton's method would be able to achieve the "optimal" complexity bounds and perform well in practice. Last but not least, our proposed trust funnel algorithm reveals the tremendous opportunity in designing practical algorithms with improved complexity bounds by combining new methods and incorporating them within a powerful framework such as *sequential quadratic programming* (SQP).

In Chapter 2, we have proposed a trust region algorithm for solving nonconvex smooth optimization problems. The important features of the algorithm are that it maintains the global and fast local convergence guarantees of a traditional trust region algorithm, but also ensures that the norm of the gradient of the objective will be reduced below a prescribed scalar constant $\epsilon \in (0, \infty)$ after at most $\mathcal{O}(\epsilon^{-3/2})$ function evaluations, gradient evaluations, or iterations. This improves upon the worst-case complexity bound for a traditional trust region algorithm, and matches that of the recently proposed ARC algorithm, which is optimal under certain conditions [16]. This complexity bound for ARC is known to be sharp, and we expect that this is also the case for TRACE. Although we have not done it in this proposal, the sharpness of this bound could be shown by verifying that

TRACE fits within the general algorithmic framework considered in [16].

For simplicity in revealing the salient features of our algorithm and its theoretical properties, we have assumed that the algorithm uses exact first- and second-order derivative information, and that each iteration involves computing a globally optimal solution of a trust region subproblem. These requirements must often be relaxed when solving large-scale problems.

In Chapter 3, we have proposed a general framework for solving smooth nonconvex optimization problems and proceeded to prove worst-case iteration complexity bounds for it. Our framework is flexible enough to cover a wide range of popular algorithms, an achievement made possible by the use of generic conditions that each trial step is required to satisfy. The use of such conditions allows for the calculation of inexact Newton steps, for example by performing minimization over expanding Krylov subspaces. Although we have presented a particular instance of our framework motivated by subproblem (3.1), additional instances can easily be derived by applying other optimization strategies for solving (3.1). Numerical experiments with an instance of our algorithm showed that it can lead to improved performance on a broad test set as compared to an implementation of a straightforward cubic regularization approach.

In Chapter 4, an algorithm has been proposed for solving equality constrained optimization problems. Based on trust funnel [38] and trust region ideas from Chapter 2, the algorithm represents a next step toward the design of practical methods for solving constrained optimization problems that offer strong worst-case iteration complexity properties. In particular, the algorithm involves two phases, the first seeking (approximate) feasibility and the second seeking optimality, where a key contribution is the fact that improvement in the objective function is sought in both phases.

5.1 Future Work

Our work proposes a new path in designing practical algorithms with good theoretical guarantees. In this dissertation, we have taken a few steps up this new path. However, the road is wide open and there are exciting opportunities ahead waiting for adventurous explorers. In this section, a few of them are discussed.

Stochastic and randomized second-order algorithms for nonconvex optimization

In the entirety of this dissertation, the function value and its first and second derivatives were assumed to be deterministic. In many real-world applications, however, this assumption seems restrictive. For example, the objective and/or models of it might be stochastic functions. Recently, the worst-case iteration complexity of first- and second-order trust region algorithms [44] and a cubic regularization method [12] have been studied. One may consider to analyze the worst-case performance of an algorithm in the `iR_Newton` framework. Such methods can be employed in solving *empirical risk minimization* (ERM) problems, arising in almost all machine learning applications. A general representation of ERM, known as the *finite sum* problem, is defined as

$$\min_{x\in\mathbb{R}^n} \; F(x) := \frac{1}{m}\sum_{i=1}^{m} f_i(x), \tag{5.1}$$

where, in machine learning (ML) applications, F is the average empirical loss function, n is the number of features, m is the number of sample points, and f_i ($i \in \{1,\dots,m\}$) is a composition of a loss and a prediction function associated with the i-th sample point.

In many ML applications, n and m are very large, which make the exact computation of the derivatives difficult. In some cases, storing the Hessian matrix may even be physically impossible. In such cases, many randomized algorithms mostly based on *stochastic gradient descent* (SGD) have been proposed. One may refer to [10] for a review of methods for solving (5.1). Recently, a randomized trust region algorithm and a randomized cubic regularization algorithm with complexity $\mathcal{O}(\epsilon^{-3/2})$ have been proposed [62].

To design a stochastic algorithm with improved worst-case complexity, the analysis or even the structure of the `iR_Newton` framework proposed in Chapter 3 needs to be revised. The algorithm must be able to deal with partial gradient and Hessian information. In addition, the fact that n is large in most ML applications demands a Hessian matrix-free algorithm, in which second-order information is only used in the form of Hessian-vector products.

The use of higher-order models

Can one improve the worst-case performance of our algorithms by using higher-order models of objective? This question has motivated new theoretical studies to improve the performance of existing algorithms. It has been shown that the worst-case iteration complexity $\mathcal{O}(\epsilon^{-(p+1)/p})$ can be obtained for some algorithms that employ p-th order Taylor models plus $(p+1)$-th order regularization terms; e.g., see [7, 8, 21, 22]. (While the theoretical behavior of these algorithms has been explored in these papers, the practical performance of such algorithms has been to be explored.) Therefore, a natural extension of this work can be considered by exploring the same possibility for our proposed algorithms.

Efficient implementation of the proposed algorithms

Our preliminary implementations of the proposed algorithms and the numerical experiments showed that these algorithms perform well in practice at least on the `CUTEr` and `CUTEst` test sets. One can possibly improve the performance of the algorithms by employing more sophisticated methods in updating the parameters especially after a rejected step (similar to the procedure proposed in [42] for the ARC algorithm), better use of memory and other computational resources.

One-phase algorithm for constrained optimization

In Chapter 4, we proposed a two-phase algorithm with improved iteration complexity which in both phases observes the objective value and the constraint violation. The proposed algorithm was shown to perform well in practice compared to a two-phase algorithm which only considers the constraint violation in the first phase. The SQP algorithm, on the other hand, is a single phase algorithm which considers to improve the objective function and the constraint violation simultaneously. Globally convergent variants of SQP, although suffering from lack of theoretical complexity bounds, perform well in practice and are considered as the state-of-the-art algorithms for constrained optimization. As mentioned, the algorithm proposed in Chapter 4 is the first step toward designing a practical—possibly single phase—algorithm with improved complexity bounds. Such an algorithm will be able to locate an approximate KKT point in a single phase procedure with improved theoretical worst-case iteration complexity.

Algorithms for general constrained nonconvex optimization

The algorithm proposed in Chapter 4 is designed to find a relative KKT point for equality constrained nonconvex optimization. In real-world problems, the constraints often contain inequalities. It is of course possible to add slack variables with proper sign to convert the inequality constraint to an equality constraint and use the proposed algorithm along with a proper projection procedure to solve the resulting problem. However, designing an algorithm with the desired worst-case complexity able to find an approximate KKT point for a generally constrained optimization might be an interesting direction for further investigation. One may first consider the case when the feasible region is a convex set. In this case, TRACE with steps projected onto the feasible region seems a viable option. More general cases, however, require further studies.

Bibliography

[1] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, Inc., Third edition, 2006.

[2] A. Beck and M. Teboulle. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[3] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Second edition, 1999.

[4] L. T. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Waanders. *Real-Time PDE-Constrained Optimization*. Society for Industrial and Applied Mathematics (SIAM), 2007.

[5] E. G. Birgin and J. M. Martínez. The Use of Quadratic Regularization with a Cubic Descent Condition for Unconstrained Optimization. *SIAM Journal on Optimization*, 27(2):1049–1074, 2017.

[6] E. G. Birgin, J. L. Gardenghi, J. M. Martínez, S. A. Santos, and Ph. L. Toint. Evaluation Complexity for Nonlinear Constrained Optimization Using Unscaled KKT Conditions and High-Order Models. *SIAM Journal on Optimization*, 26(2):951–967, 2016.

[7] E. G. Birgin, J. L. Gardenghi, J. M. Martínez, and S. A. Santos. On the Use of Third-Order Models with Fourth-Order Regularization for Unconstrained Optimization. 2017. URL http://www.optimization-online.org/DB_HTML/2017/11/6324.html.

[8] 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. *Mathematical Programming*, 163(1):359–368, 2017.

[9] G. Biros and O. Ghattas. Parallel Lagrange–Newton–Krylov–Schur Methods for PDE-Constrained Optimization. Part I: The Krylov–Schur Solver. *SIAM Journal on Scientific Computing*, 27(2):687–713, 2005.

[10] L. Bottou, F. Curtis, and J. Nocedal. Optimization Methods for Large-Scale Machine Learning. *SIAM Review*, 60(2):223–311, 2018.

[11] R. H. Byrd, F. E. Curtis, and J. Nocedal. An Inexact SQP Method for Equality Constrained Optimization. *SIAM Journal on Optimization*, 19(1):351–369, 2008.

[12] C. Cartis and K. Scheinberg. Global Convergence Rate Analysis of Unconstrained Optimization Methods Based on Probabilistic Models. *Mathematical Programming*, 169(2):337–375, 2018.

[13] 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 Journal on Optimization*, 20(6):2833–2852, 2010.

[14] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Adaptive Cubic Regularisation Methods for Unconstrained Optimization. Part I: Motivation, Convergence and Numerical Results. *Mathematical Programming*, 127(2):245–295, 2011.

[15] 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. *Mathematical Programming*, 130(2):295–319, 2011.

[16] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Optimal Newton-type Methods for Nonconvex Smooth Optimization Problems. Technical Report ERGO Technical Report 11-009, School of Mathematics, University of Edinburgh, 2011.

[17] C. Cartis, N. I. M. Gould, and Ph. L. Toint. On the Evaluation Complexity of Composite Function Minimization with Applications to Nonconvex Nonlinear Programming. *SIAM Journal on Optimization*, 21(4):1721–1739, 2011.

[18] C. Cartis, N. I. M. Gould, and Ph. L. Toint. An Adaptive Cubic Regularization Algorithm for Nonconvex Optimization with Convex Constraints and its Function-Evaluation Complexity. *IMA Journal of Numerical Analysis*, 32(4):1662–1695, 2012.

[19] C. Cartis, N. I. M. Gould, and Ph. L. Toint. On the Evaluation Complexity of Cubic Regularization Methods for Potentially Rank-Deficient Nonlinear Least-Squares Problems and its Relevance to Constrained Nonlinear Optimization. *SIAM Journal on Optimization*, 23(3):1553–1574, 2013.

[20] C. Cartis, N. I. M. Gould, and Ph. L. Toint. On the Complexity of Finding First-Order Critical Points in Constrained Nonlinear Programming. *Mathematical Programming*, 144(1):93–106, 2014.

[21] C. Cartis, N. Gould, and Ph. L. Toint. Evaluation Complexity Bounds for Smooth Constrained Nonlinear Optimisation Using Scaled KKT Conditions, High-Order Models and the Criticality Measure χ. arXiv:1705.04895, 2017. URL https://arxiv.org/pdf/1705.04895.pdf.

[22] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Second-Order Optimality and Beyond: Characterization and Evaluation Complexity in Convexly Constrained Nonlinear Optimization. *Foundations of Computational Mathematics*, 18(5):1073–1107, 2018.

[23] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. Society for Industrial and Applied Mathematics (SIAM), 2000.

[24] L. Cui, W. Kuo, H. T. Loh, and M. Xie. Optimal Allocation of Minimal & Perfect Repairs under Resource Constraints. *IEEE Transactions on Reliability*, 53(2):193–199, 2004.

[25] F. E. Curtis, J. Nocedal, and A. Wächter. A Matrix-Free Algorithm for Equality Constrained Optimization Problems with Rank-Deficient Jacobians. *SIAM Journal on Optimization*, 20(3):1224–1249, 2009.

[26] F. E. Curtis, T. C. Johnson, D .P. Robinson, and A. Wächter. An Inexact Sequential Quadratic Optimization Algorithm for Nonlinear Optimization. *SIAM Journal on Optimization*, 24(3):1041–1074, 2014.

[27] F. E. Curtis, N. I. M. Gould, D. P. Robinson, and Ph. L. Toint. An Interior-Point Trust-Funnel Algorithm for Nonlinear Optimization. *Mathematical Programming*, 161 (1):73–134, 2017.

[28] F. E. Curtis, D. P. Robinson, and M. Samadi. A Trust Region Algorithm with a Worst-Case Iteration Complexity of $O(\epsilon^{-3/2})$ for Nonconvex Optimization. *Mathematical Programming*, 162(1):1–32, 2017.

[29] F. E. Curtis, D. P. Robinson, and M. Samadi. Complexity Analysis of a Trust Funnel Algorithm for Equality Constrained Optimization. *SIAM Journal on Optimization*, 28(2):1533–1563, 2018.

[30] 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. *Accepted for publication in the IMA Journal of Numerical Analysis*, 2018. doi: 10.1093/imanum/dry022. URL http://dx.doi.org/10.1093/imanum/dry022.

[31] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Society for Industrial and Applied Mathematics (SIAM), 1996.

[32] E. D. Dolan and J. J. Moré. Benchmarking Optimization Software with Performance Profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[33] J.-P. Dussault. ARCq: A New Adaptive Regularization by Cubics. *Optimization Methods and Software*, 33(2):322–335, 2018.

[34] J.-P. Dussault and D. Orban. Scalable Adaptive Cubic Regularization Methods. Technical Report G-2015-109, GERAD, 2017.

[35] N. I. M. Gould and D. P. Robinson. A Second Derivative SQP Method: Global Convergence. *SIAM Journal on Optimization*, 20(4):2023–2048, 2010.

[36] N. I. M. Gould and D. P. Robinson. A Second Derivative SQP Method: Local Convergence and Practical Issues. *SIAM Journal on Optimization*, 20(4):2049–2079, 2010.

[37] N. I. M. Gould and D. P. Robinson. A Second Derivative SQP Method with a "Trust-Region-Free" Predictor Step. *IMA Journal of Numerical Analysis*, 32(2):580–601, 2012.

[38] N. I. M. Gould and Ph. L. Toint. Nonlinear Programming Without a Penalty Function or a Filter. *Mathematical Programming*, 122(1):155–196, 2010.

[39] N. I. M. Gould, D. Orban, and Ph. L. Toint. GALAHAD, a Library of Thread-Safe Fortran 90 Packages for Large-Scale Nonlinear Optimization. *ACM Transactions on Mathematical Software (TOMS)*, 29(4):353–372, 2003.

[40] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr and SifDec: A Constrained and Unconstrained Testing Environment, Revisited. *ACM Transactions on Mathematical Software (TOMS)*, 29(4):373–394, 2003.

[41] N. I. M. Gould, D. P. Robinson, and H. S. Thorne. On Solving Trust-Region and other Regularised Subproblems in Optimization. *Mathematical Programming Computation*, 2(1):21–57, 2010.

[42] N. I. M. Gould, M. Porcelli, and P. L. Toint. Updating the Regularization Parameter in the Adaptive Cubic Regularization Algorithm. *Computational Optimization and Applications*, 53(1):1–22, 2012.

[43] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEst: A Constrained and Unconstrained Testing Environment with Safe Threads. Technical Report RAL-TR-2013-005, Rutherford Appleton Laboratory, 2013.

[44] S. Gratton, C. W. Royer, L. N. Vicente, and Z. Zhang. Complexity and Global Rates of Trust-Region Methods Based on Probabilistic Models. *IMA Journal of Numerical Analysis*, 38(3):1579–1597, 2018.

[45] 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.

[46] I. Griva, S. G. Nash, and A. Sofer. *Linear and Nonlinear Optimization*. Society for Industrial and Applied Mathematics (SIAM), Second edition, 2008.

[47] R. J. Hathaway. A Constrained Formulation of Maximum-Likelihood Estimation for Normal Mixture Distributions. *The Annals of Statistics*, 13(2):795–800, 1985.

[48] N.-S. Hsu and K.-W. Cheng. Network Flow Optimization Model for Basin-Scale Water Supply Planning. *Journal of Water Resources Planning and Management*, 128 (2):102–112, 2002.

[49] P. Marti, C. Lin, S. A. Brandt, M. Velasco, and J. M. Fuertes. Optimal State Feedback Based Resource Allocation for Resource-Constrained Control Tasks. In *Real-Time Systems Symposium, 2004. Proceedings. 25th IEEE International*, pages 161–172. IEEE, 2004.

[50] J. M. Martínez. On High-Order Model Regularization for Constrained Optimization. *SIAM Journal on Optimization*, 27(4):2447–2458, 2017.

[51] J. L. Morales, J. Nocedal, and Y. Wu. A Sequential Quadratic Programming Algorithm with an Additional Equality Constrained Phase. *IMA Journal of Numerical Analysis*, 32(2):553–579, 2011.

[52] J. J. Moré and D. C. Sorensen. Computing a Trust Region Step. *SIAM Journal on Scientific and Statistical Computing*, 4(3):553–572, 1983.

[53] Yu. Nesterov. A Method of Solving a Convex Programming Problem with Convergence Rate $\mathcal{O}(1/k^2)$. *Soviet Mathematics Doklady*, 27(2), 1983.

[54] Yu. Nesterov. *Introductory Lectures on Convex Optimization*, volume 87. Springer Science & Business Media, 2004.

[55] Yu. Nesterov. Accelerating the Cubic Regularization of Newton's Method on Convex Problems. *Mathematical Programming*, 112(1):159–181, 2007.

[56] Yu. Nesterov and B. T. Polyak. Cubic Regularization of Newton's Method and its Global Performance. *Mathematical Programming*, 108(1):117–205, 2006.

[57] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, Second edition, 2006.

[58] K. E. Nygard, P. R. Chandler, and M. Pachter. Dynamic Network Flow Optimization Models for Air Vehicle Resource Allocation. In *American Control Conference, 2001. Proceedings of the 2001*, volume 3, pages 1853–1858. IEEE, 2001.

[59] T. Rees, H. S. Dollar, and A. J. Wathen. Optimal Solvers for PDE-Constrained Optimization. *SIAM Journal on Scientific Computing*, 32(1):271–298, 2010.

[60] A. Ruszczynski. *Nonlinear Optimization*. Princeton University Press, 2006.

[61] M. Weiser, P. Deuflhard, and B. Erdmann. Affine Conjugate Adaptive Newton Methods for Nonlinear Elastomechanics. *Optimization Methods and Software*, 22(3):413–431, 2007.

[62] P. Xu, F. Roosta-Khorasani, and W. M. Mahoney. Newton-Type Methods for Non-Convex Optimization Under Inexact Hessian Information. arXiv:1708.07164, 2018. URL https://arxiv.org/abs/1708.07164.

Appendix A

Subproblem Solver for TRACE

The TRACE algorithm follows the framework of a trust region algorithm in which, in each iteration, a subproblem with a quadratic objective and a trust region constraint is solved to optimality and all other steps are explicit computations involving the iterate and related sequences. The only exception is Step 13 in which a new trust region radius is computed via the CONTRACT subroutine. In this appendix, we outline a practical procedure entailing the main computational components of CONTRACT, revealing that it can be implemented in such a way that each iteration of TRACE need not be more computational expensive than similar implementations of those in a traditional trust region algorithm or ARC.

Suppose that Steps 3 and 19 are implemented using a traditional approach of applying Newton's method to solve a secular equation of the form $\phi_k(\lambda) = 0$, where, for a given $\lambda \geq \max\{0, -\xi_{k,1}\}$ (where, as in the proof of Lemma 11, we define $\xi_{k,1}$ as the leftmost eigenvalue of H_k), the vector $s_k(\lambda)$ is defined as a solution of the linear system (2.2) and

$$\phi_k(\lambda) = \|s_k(\lambda)\|_2^{-1} - \delta_k^{-1}. \tag{A.1}$$

A practical implementation of such an approach involves the initialization and update of an interval of uncertainty, say $[\underline{\lambda}, \overline{\lambda}]$, in which the dual variable λ_k corresponding to a solution $s_k = s_k(\lambda_k)$ of $\mathcal{Q}_k$ is known to lie [52]. In particular, for a given estimate $\lambda \in [\underline{\lambda}, \overline{\lambda}]$, a factorization of $(H_k + \lambda I)$ is computed (or at least attempted), yielding a trial solution $s_k(\lambda)$ and a corresponding derivative of ϕ_k for the application of a (safeguarded) Newton iteration.

In the context of such a strategy for the implementation of Steps 3 and 19, most

of the computations involved in the CONTRACT subroutine can be considered as part of the initialization process for such a Newton iteration, if not a *replacement* for the entire Newton iteration. For example, if Step 33 is reached, then the computation of (λ, s) in Steps 33–34 are exactly those that would be performed in such a Newton iteration with an initial solution estimate of $\lambda \leftarrow \gamma_\lambda \lambda_k$. If Step 36 is reached, then the solution (s_{k+1}, λ_{k+1}) of $\mathcal{Q}_{k+1}$ in Step 19 is yielded by this computation and a Newton solve of a secular equation is not required; otherwise, if Step 38 is reached, then one could employ $\overline{\lambda} \leftarrow \lambda$ in the Newton iteration for solving $\mathcal{Q}_{k+1}$ in Step 19. Overall, if Step 33 is reached, then the computations in CONTRACT combined with Step 19 are no more expensive than the subproblem solve in a traditional trust region algorithm, and may be significantly cheaper in cases when Step 36 is reached.

The situation is similar when Step 25 is reached. In particular, if the pair (λ, s) computed in Steps 25–26 result in Step 28 being reached, then the pair (s_{k+1}, λ_{k+1}) required in Step 19 is available without having to run an expensive Newton iteration to solve a secular equation, meaning that computational expense is saved in our mechanism for setting δ_{k+1} implicitly via our choice of the dual variable λ_{k+1}. On the other hand, if Step 30 is reached, then the algorithm requests a value $\lambda \in (\lambda_k, \hat{\lambda})$ such that $\underline{\sigma} \leq \lambda / \|s_k(\lambda)\|_2 \leq \overline{\sigma}$. A variety of techniques could be employed for finding such a λ, but perhaps the most direct is to consider a technique such as [14, Algorithm 6.1] in which a cubic regularization subproblem is solved using a (safeguarded) Newton iteration applied to solve a secular equation similar to (A.1). It should be noted, however, that while [14, Algorithm 6.1] attempts to solve $\lambda / \|s_k(\lambda)\|_2 = \sigma$ for some given $\sigma > 0$, the computation in Step 30 merely requires $\underline{\sigma} \leq \lambda / \|s_k(\lambda)\|_2 \leq \overline{\sigma}$, meaning that one could, say, choose $\sigma = (\underline{\sigma} + \overline{\sigma})/2$, but terminate the Newton iteration as soon as $\lambda / \|s_k(\lambda)\|_2$ is computed in the (potentially very large) interval $[\underline{\sigma}, \overline{\sigma}]$. Clearly, such a computation is no more expensive than [14, Algorithm 6.1].

Finally, it is worthwhile to note that since the CONTRACT subroutine desires the computation of a trust region radius such that the new corresponding dual variable satisfies $\lambda_{k+1} > \lambda_k \geq \max\{0, -\xi_{k,1}\}$, it follows that, after a contraction, the subproblem $\mathcal{Q}_{k+1}$ will not involve the well known "hard case" in the context of solving a trust region subproblem. (We remark that this avoidance of the "hard case" does not necessarily occur if one were to perform a contraction merely by setting the trust region radius as a fraction of the norm of the trial step, as is typically done in other trust region methods.)

Appendix B

Subproblem Solution Properties and Numerical Results for `iR_Newton`

B.1 Subproblem Solution Properties

In this appendix, we explore properties of any first-order stationary solution of problem $\mathcal{P}_k(\sigma_k^{\mathrm{L}}, \sigma_k^{\mathrm{U}})$ defined as (3.1). Let us define a Lagrangian function for (3.1) as

$$\begin{aligned}\mathcal{L}(s, \lambda, \beta^{\mathrm{L}}, \beta^{\mathrm{U}}, \beta^{\mathrm{N}}) = f_k + g_k^T s + \tfrac{1}{2} s^T (H_k + \lambda I) s \\ - \tfrac{\beta^{\mathrm{L}}}{2}(\lambda^2 - (\sigma_k^{\mathrm{L}})^2 \|s\|^2) + \tfrac{\beta^{\mathrm{U}}}{2}(\lambda^2 - (\sigma_k^{\mathrm{U}})^2 \|s\|^2) - \beta^{\mathrm{N}} \lambda,\end{aligned}$$

where $(\beta^{\mathrm{L}}, \beta^{\mathrm{U}}) \in \mathbb{R}_+ \times \mathbb{R}_+$ are the dual variables associated with the left-hand and right-hand constraints on λ, respectively, and $\beta^{\mathrm{N}} \in \mathbb{R}_+$ is the dual variable associated with the nonnegativity constraint on λ. The tuple $(s_k, \lambda_k, \beta_k^{\mathrm{L}}, \beta_k^{\mathrm{U}}, \beta_k^{\mathrm{N}})$ is a first-order primal-dual stationary solution of $\mathcal{P}_k(\sigma_k^{\mathrm{L}}, \sigma_k^{\mathrm{U}})$ if it satisfies the following conditions:

$$g_k + (H_k + \lambda_k I) s_k + \beta_k^{\mathrm{L}} (\sigma_k^{\mathrm{L}})^2 s_k - \beta_k^{\mathrm{U}} (\sigma_k^{\mathrm{U}})^2 s_k = 0, \tag{B.1a}$$

$$\tfrac{1}{2}\|s_k\|^2 - \lambda_k (\beta_k^{\mathrm{L}} - \beta_k^{\mathrm{U}}) - \beta_k^{\mathrm{N}} = 0, \tag{B.1b}$$

$$0 \le \beta_k^{\mathrm{L}} \perp (\lambda_k^2 - (\sigma_k^{\mathrm{L}})^2 \|s_k\|^2) \ge 0, \tag{B.1c}$$

$$0 \le \beta_k^{\mathrm{U}} \perp (\lambda_k^2 - (\sigma_k^{\mathrm{U}})^2 \|s_k\|^2) \le 0, \quad \text{and} \tag{B.1d}$$

$$0 \leq \beta_k^{\mathrm{N}} \perp \lambda_k \geq 0. \tag{B.1e}$$

We make the following assumption throughout this appendix.

Assumption 16. *The vector g_k is nonzero.*

Under this assumption, the following lemma is a simple consequence of (B.1a).

Lemma 77. *Any solution of (3.1) has $s_k \neq 0$.*

We now establish conditions that must hold depending on the value of $\sigma_k^{\mathrm{L}} \in \mathbb{R}_+$.

Lemma 78. *The following hold true for any solution of (B.1).*

(i) If $\sigma_k^{\mathrm{L}} > 0$, then $\lambda_k > 0$, $\beta_k^{\mathrm{N}} = 0$, $\beta_k^{\mathrm{L}} > 0$, and $\lambda_k = \sigma_k^{\mathrm{L}} \|s_k\|$.

(ii) If $\sigma_k^{\mathrm{L}} = 0$, then $\lambda_k = 0$.

Proof. Consider part (i). For the sake of deriving a contradiction, suppose $\sigma_k^{\mathrm{L}} > 0$ and $\lambda_k = 0$. These, along with Lemma 77, imply that $0 = \lambda_k^2 < (\sigma_k^{\mathrm{L}})^2 \|s_k\|^2$, which contradicts (B.1c). Hence, $\lambda_k > 0$, as claimed. Then, it follows from (B.1e) that $\beta_k^{\mathrm{N}} = 0$, as claimed. Next, observe that from (B.1b), Lemma 77, $\beta_k^{\mathrm{N}} = 0$, $\lambda_k > 0$, and $(\beta_k^{\mathrm{L}}, \beta_k^{\mathrm{U}}) \geq 0$, it follows that $\beta_k^{\mathrm{L}} > 0$, as claimed. This, along with (B.1c), implies that $\lambda_k^2 = (\sigma_k^{\mathrm{L}})^2 \|s_k\|^2$. This implies that $\lambda_k = \pm(\sigma_k^{\mathrm{L}}) \|s_k\|$, which combined with $\lambda_k \in \mathbb{R}^+$ means that $\lambda_k = \sigma_k^{\mathrm{L}} \|s_k\|$, as claimed.

Now consider part (ii). For the sake of deriving a contradiction, suppose that $\sigma_k^{\mathrm{L}} = 0$ and $\lambda_k > 0$. Then, it follows from (B.1e) that $\beta_k^{\mathrm{N}} = 0$. Moreover, combining $\sigma_k^{\mathrm{L}} = 0$ and $\lambda_k > 0$, it follows from (B.1c) that $\beta_k^{\mathrm{L}} = 0$. It now follows from $\beta_k^{\mathrm{L}} = 0$, $\beta_k^{\mathrm{N}} = 0$, and (B.1b) that

$$\tfrac{1}{2} \|s_k\|^2 = -\lambda_k \beta_k^{\mathrm{U}} \leq 0, \tag{B.2}$$

where the inequality follows from $\lambda_k > 0$ and $\beta_k^{\mathrm{U}} \geq 0$. This contradicts Lemma 77. □

Our main result is the following. In part (i) with $\sigma_k^{\mathrm{L}} > 0$, we show that solving (3.1) is equivalent to solving what may be referred to as an ARC subproblem [14]. In part (ii) with $\sigma_k^{\mathrm{L}} = 0$, we show that it is equivalent to minimizing a quadratic, if a minimizer exists.

Theorem 15. *The following hold true.*

(i) Suppose $\sigma_k^{\mathrm{L}} > 0$. *Then,* (3.1) *has a solution* (s_k, λ_k), *which can be obtained as*

$$s_k \in \arg\min_{s \in \mathbb{R}^n} \ (f_k + g_k^T s + \tfrac{1}{2} s^T H_k s + \tfrac{1}{2}\sigma_k^{\mathrm{L}} \|s\|^3), \tag{B.3}$$

then setting $\lambda_k = \sigma_k^{\mathrm{L}} \|s_k\| > 0$.

(ii) If $\sigma_k^{\mathrm{L}} = 0$, *then a solution of problem* (3.1) *exists if and only if* $H_k \succeq 0$ *and* $g_k^T u = 0$ *for all* $u \in \mathrm{Null}(H_k)$. *In such cases, computing a solution* (s_k, λ_k) *of problem* (3.1) *is equivalent to computing a solution* s_k *of problem* (3.20) *and setting* $\lambda_k = 0$.

Proof. Consider part (i). Since $\sigma_k^{\mathrm{L}} > 0$, it follows from Lemma 78 that problem (3.1) is equivalent to

$$\begin{aligned} \min_{(s,\lambda) \in \mathbb{R}^n \times \mathbb{R}_+} \quad & f_k + g_k^T s + \tfrac{1}{2} s^T (H_k + \lambda I) s \\ \text{s.t.} \quad & \sigma_k^{\mathrm{L}} \|s\| = \lambda, \end{aligned} \tag{B.4}$$

where, by Lemma 77, it follows that the solution has $\lambda_k > 0$, as desired. Substituting the constraint of (B.4) into the objective of (B.4), one finds that solving it is equivalent to solving (B.3) for s_k, then setting $\lambda_k = \sigma_k^{\mathrm{L}} \|s_k\|$, as claimed. Since $\sigma_k^{\mathrm{L}} > 0$, a minimizer of problem (B.3) exists because it involves the minimization of a coercive function.

Now consider part (ii). Since $\sigma_k^{\mathrm{L}} = 0$, it follows from Lemma 78 that $\lambda_k = 0$, meaning that problem (3.1) is equivalent to (3.20). This problem has a solution if and only if the objective is bounded below, which is the case if and only if $H_k \succeq 0$ and $g_k^T u = 0$ for all $u \in \mathrm{Null}(H_k)$. □

B.2 Subproblem Solution Properties Over Subspaces

In this appendix, we explore properties of any first-order stationary solution (when one exists) of problem $\mathcal{P}_k(\sigma_k^{\mathrm{L}}, \sigma_k^{\mathrm{U}})$ defined as (3.1) when the search space for s is restricted to a subspace of $\mathbb{R}^n$. Specifically, for some m-dimensional subspace $\mathcal{L}_k \subseteq \mathbb{R}^n$, consider the problem

$$\begin{aligned} \min_{(s,\lambda) \in \mathcal{L}_k \times \mathbb{R}_+} \quad & f_k + g_k^T s + \tfrac{1}{2} s^T (H_k + \lambda I) s \\ \text{s.t.} \quad & (\sigma_k^{\mathrm{L}})^2 \|s\|^2 \leq \lambda^2 \leq (\sigma_k^{\mathrm{U}})^2 \|s\|^2. \end{aligned} \tag{B.5}$$

Given an orthogonal basis R_k for $\mathcal{L}_k$, a solution of (B.5) can be obtained from that of

$$\begin{aligned} \min_{(v,\lambda)\in\mathbb{R}^m\times\mathbb{R}_+} \quad & f_k + g_k^T R_k v + \tfrac{1}{2}(R_k v)^T (H_k + \lambda I) R_k v \\ \text{s.t.} \quad & (\sigma_k^{\mathrm{L}})^2 \|v\|^2 \le \lambda^2 \le (\sigma_k^{\mathrm{U}})^2 \|v\|^2. \end{aligned} \tag{B.6}$$

Specifically, if $(v_k, \lambda_k, \beta_k^{\mathrm{L}}, \beta_k^{\mathrm{U}}, \beta_k^{\mathrm{N}})$ is a first-order primal-dual solution of problem (B.6), then the tuple $(s_k, \lambda_k, \beta_k^{\mathrm{L}}, \beta_k^{\mathrm{U}}, \beta_k^{\mathrm{N}})$ with $s_k = R_k v_k$ is such a solution of problem (B.5).

In Appendix B.1, we proved properties of a solution (if one exists) of a problem of the form (B.6). Let us now translate the results of that appendix to the present setting, for which we require the following assumption on the reduced gradient $R_k^T g_k$.

Assumption 17. *The vector $R_k^T g_k$ is nonzero.*

Lemma 79. *Any solution of (B.6) has $v_k \neq 0$.*

Lemma 80. *The following hold for any first-order primal-dual solution of (B.5).*

(i) If $\sigma_k^{\mathrm{L}} > 0$, then $\lambda_k > 0$, $\beta_k^{\mathrm{N}} = 0$, $\beta_k^{\mathrm{L}} > 0$, and $\lambda_k = \sigma_k^{\mathrm{L}} \|v_k\|$.

(ii) If $\sigma_k^{\mathrm{L}} = 0$, then $\lambda_k = 0$.

Theorem 16. *The following hold true.*

(i) Suppose $\sigma_k^{\mathrm{L}} > 0$. Then, (B.6) has a solution (v_k, λ_k), which can be obtained as

$$v_k \in \arg\min_{v\in\mathbb{R}^m} \ (f_k + g_k^T R_k v + \tfrac{1}{2} v^T R_k^T H_k R_k v + \tfrac{1}{2}\sigma_k^{\mathrm{L}} \|v\|^3), \tag{B.7}$$

then setting $\lambda_k = \sigma_k^{\mathrm{L}} \|v_k\| > 0$.

(ii) If $\sigma_k^{\mathrm{L}} = 0$, then a solution of (B.6) exists if and only if $R_k^T H_k R_k \succeq 0$ and $g_k^T R_k u = 0$ for all $u \in \mathrm{Null}(R_k^T H_k R_k)$. In such cases, computing a solution (v_k, λ_k) of problem (B.6) is equivalent to computing a solution v_k of

$$\min_{v\in\mathbb{R}^m} \ f_k + g_k^T R_k v + \tfrac{1}{2} v^T R_k^T H_k R_k v \tag{B.8}$$

and setting $\lambda_k = 0$.

Considering problem (B.7), we obtain the following result from [14, Lemma 3.2].

Lemma 81. *If $\sigma_k^{\text{L}} > 0$, then v_k from (B.7) satisfies*

$$g_k^T R_k v_k + v_k^T R_k^T H_k R_k v_k + \tfrac{3}{2}\sigma_k^{\text{L}} \|v_k\|^3 = 0 \tag{B.9a}$$

$$v_k^T R_k^T H_k R_k v_k + \tfrac{3}{2}\sigma_k^{\text{L}} \|v_k\|^3 \geq 0 \tag{B.9b}$$

$$R_k^T H_k R_k + \tfrac{3}{2}\sigma_k^{\text{L}} \|v_k\| I \succeq 0. \tag{B.9c}$$

We now show that, under certain reasonable assumptions, solutions of the primal-dual reduced-space subproblem (B.5) satisfy the conditions required by Assumptions 7 and 11.

Theorem 17. *The following hold true.*

(a) Any solution of problem (B.5) satisfies (3.3b).

(b) Any solution of problem (B.5) satisfies (3.3a) provided $g_k \in \mathcal{L}_k$.

(c) Any solution of problem (B.5) satisfies (3.3c) provided $\mathcal{L}_k = \mathbb{R}^n$.

(d) Any solution of problem (B.5) satisfies (3.17) for any $\kappa_4 \geq \frac{3}{2} \sup_{k \in \mathbb{N}_+} \{\sigma_k^{\text{L}}\}$.

Proof. Any first-order primal-dual solution $(s_k, \lambda_k, \beta_k^{\text{L}}, \beta_k^{\text{U}}, \beta_k^{\text{N}})$ of problem (B.5) corresponds to such a solution $(v_k, \lambda_k, \beta_k^{\text{L}}, \beta_k^{\text{U}}, \beta_k^{\text{N}})$ of problem (B.6) where $s_k = R_k v_k$. Hence, throughout this proof, for any solution vector s_k for problem (B.5), we may let $s_k = R_k v_k$ where v_k satisfies the properties in Lemmas 79–81.

First, suppose $\sigma_k^{\text{L}} > 0$, which by Theorem 16(i) implies that problem (B.5) has a solution. Then, it follows from (B.9a), $s_k = R_k v_k$, and Lemma 80(i) that

$$0 = g_k^T s_k + s_k^T H_k s_k + \tfrac{3}{2}\sigma_k^{\text{L}} \|s_k\|^3 = g_k^T s_k + s_k^T H_k s_k + \tfrac{3}{2}\lambda_k \|s_k\|^2,$$

which means that

$$s_k^T (g_k + (H_k + \lambda_k I) s_k) = -\tfrac{1}{2}\lambda_k \|s_k\|^2. \tag{B.10}$$

Meanwhile, from (B.9b), $s_k = R_k v_k$, and Lemma 80(i), it follows that

$$0 \leq s_k^T H_k s_k + \tfrac{3}{2}\sigma_k^{\text{L}} \|s_k\|^3 = s_k^T H_k s_k + \tfrac{3}{2}\lambda_k \|s_k\|^2 = s_k^T (H_k + \lambda_k I) s_k + \tfrac{1}{2}\lambda_k \|s_k\|^2,$$

which means that

$$-\tfrac{1}{4}\lambda_k \|s_k\|^2 \leq \tfrac{1}{2} s_k^T (H_k + \lambda_k I) s_k. \tag{B.11}$$

It follows from (B.10), (B.11), $\lambda_k > 0$ (by Lemma 80(i)), and $(\kappa_1, \kappa_2) \in \mathbb{R}_{++} \times \mathbb{R}_{++}$ that

$$\begin{aligned} s_k^T(g_k + (H_k + \lambda_k I)s_k) = -\tfrac{1}{2}\lambda_k\|s_k\|^2 &\leq \min\{\tfrac{1}{2}\kappa_1\|s_k\|^2, \tfrac{1}{2}s_k^T(H_k + \lambda_k I)s_k - \tfrac{1}{4}\lambda_k\|s_k\|^2\} \\ &\leq \min\{\tfrac{1}{2}\kappa_1\|s_k\|^2, \tfrac{1}{2}s_k^T(H_k + \lambda_k I)s_k + \tfrac{1}{2}\kappa_2\|s\|^3\}, \end{aligned}$$

which implies (3.3b). This establishes that part (a) is true. Now consider part (b). From Theorem 16, [14, Lemma 2.1], and $s_k = R_k v_k$, it follows that

$$f_k - q_k(s_k) - \tfrac{1}{2}\sigma_k^{\text{L}}\|s_k\|^3 \geq \frac{\|R_k^T g_k\|}{6\sqrt{2}} \min\left\{ \frac{\|R_k^T g_k\|}{1 + \|R_k^T H_k R_k\|}, \frac{1}{\sqrt{6}}\sqrt{\frac{\|R_k^T g_k\|}{\sigma_k^{\text{L}}}} \right\}.$$

Since, under assumption, $g_k \in \mathcal{L}_k$ so that $g_k = R_k y$ for some $y \in \mathbb{R}^m$, it follows that

$$\|R_k^T g_k\| = \|R_k^T R_k y\| = \|y\| = \|R_k y\| = \|g_k\|.$$

Combining this with $\|R_k^T H_k R_k\| \leq \|H_k\|$ and the previous displayed inequality shows

$$f_k - q_k(s_k) - \tfrac{1}{2}\sigma_k^{\text{L}}\|s_k\|^3 \geq \frac{\|g_k\|}{6\sqrt{2}} \min\left\{ \frac{\|g_k\|}{1 + \|H_k\|}, \frac{1}{\sqrt{6}}\sqrt{\frac{\|g_k\|}{\sigma_k^{\text{L}}}} \right\}.$$

This may now be combined with Theorem 16 (specifically $\lambda_k = \sigma_k^{\text{L}}\|s_k\| > 0$) to obtain

$$f_k - q_k(s_k) \geq f_k - q_k(s_k) - \tfrac{1}{2}\sigma_k^{\text{L}}\|s_k\|^3 \geq \frac{\|g_k\|}{6\sqrt{2}} \min\left\{ \frac{\|g_k\|}{1 + \|H_k\|}, \frac{1}{\sqrt{6}}\sqrt{\frac{\|g_k\|\|s_k\|}{\lambda_k}} \right\},$$

which means that (s_k, λ_k) satisfies (3.3a), proving part (b). Now consider part (c). It follows from Theorem 15(i) and the optimality conditions for problem (B.3) that

$$0 = g_k + H_k s_k + \tfrac{3}{2}\sigma_k^{\text{L}}\|s_k\|s_k = g_k + H_k s_k + \tfrac{3}{2}\lambda_k s_k = g_k + (H_k + \lambda_k I)s_k + \tfrac{1}{2}\lambda_k s_k.$$

This and the fact that $\kappa_3 > 0$ imply that

$$\|g_k + (H_k + \lambda_k I)s_k\| = \tfrac{1}{2}\lambda_k\|s_k\| \leq \lambda_k\|s_k\| + \kappa_3\|s_k\|^2,$$

which completes the proof of part (c). Finally, consider part (d). From (B.9c), the fact

that $\|s_k\| = \|v_k\|$, and $\kappa_4 \geq \frac{3}{2} \sup_{k \in \mathbb{N}_+} \{\sigma_k^{\text{L}}\}$, it follows that

$$\xi(R_k^T H_k R_k) \geq -\tfrac{3}{2}\sigma_k^{\text{L}} \|s_k\| \geq -\kappa_4 \|s_k\|,$$

as desired to prove part (d).

Now suppose that $\sigma_k^{\text{L}} = 0$. From Theorem 16(ii), a solution of problem (B.5) exists if and only if $R_k^T H_k R_k \succeq 0$ and $g_k^T R_k u = 0$ for all $u \in \text{Null}(R_k^T H_k R_k)$. If this is not the case, then there is nothing left to prove; hence, let us assume that these conditions hold. From these conditions, Theorem 16(ii), the optimality conditions of problem (B.8), the fact that $\lambda_k = 0$, and $s_k = R_k v_k$, it follows that

$$g_k^T s_k + s_k^T H_k s_k = 0 \quad \text{and} \quad s_k^T H_k s_k \geq 0.$$

This shows that (3.3b) holds, proving part (a) for this case. Next, since v_k is given by the solution of problem (B.8), it follows that the reduction in the objective yielded by v_k is at least as large as the reduction obtained by minimizing the objective over the span of $-R_k^T g_k$. Hence, from standard theory on Cauchy decrease (see [23] or [57]), one can conclude that

$$f_k - q_k(s_k) \geq \frac{\|R_k^T g_k\|}{2} \min \left\{ \frac{\|R_k^T g_k\|}{1 + \|R_k^T H_k R_k\|}, \|s_k\| \right\}.$$

Thus, using the arguments in the previous paragraph under the assumption that $g_k \in \mathcal{L}_k$, one is led to the conclusion that (3.3a) holds, which proves part (b) for this case. Next, when $\mathcal{L}_k = \mathbb{R}^n$, the optimality conditions for problem (B.8) imply that $g_k + H_k s_k = 0$, which, since $\lambda_k = 0$, implies that (3.3c) holds, proving part (c). Finally, since $R_k^T H_k R_k \succeq 0$, it follows that (3.17) holds, proving part (d). □

B.3 Detailed Numerical Results

Further details of the results of our numerical experiments are shown in Table B.1. In the table, `#Var` indicates the number of variables, `#Iter` indicates the number of iterations required (with `%Newton` indicating the percentage that were inexact Newton steps with $\lambda_k = 0$), `#Acc` indicates the number of accepted steps (again with `%Newton` indicating the percentage that were inexact Newton steps), `#Hv-prod` indicates the number of Hessian-vector products required, and `#T-fact` indicates the number of tridiagonal matrix

factorizations required.

Table B.1: Numerical results for `iARC` and `iR_Newton`.

Prob	#Var	Alg	#Iter (%Newton)	#Acc (%Newton)	#Hv-prod	#T-fact
AKIVA	2	iARC	5	5	15	20
		iR_Newton	5 (%100)	5 (%100)	10	0
ALLINITU	4	iARC	11	8	56	61
		iR_Newton	8 (%50)	6 (%67)	25	21
ARGLINA	200	iARC	3	3	6	3
		iR_Newton	1 (%100)	1 (%100)	1	0
ARGLINB	200	iARC	2	2	4	2
		iR_Newton	1 (%100)	1 (%100)	1	0
ARWHEAD	5000	iARC	4	4	10	6
		iR_Newton	4 (%100)	4 (%100)	5	0
BARD	3	iARC	11	8	50	52
		iR_Newton	11 (%91)	10 (%90)	28	6
BDQRTIC	5000	iARC	9	9	34	33
		iR_Newton	9 (%100)	9 (%100)	17	0
BEALE	2	iARC	11	8	33	47
		iR_Newton	12 (%42)	8 (%62)	29	29
BIGGS6	6	iARC	457	383	3446	3679
		iR_Newton	419 (%96)	406 (%97)	1557	183
BOX	10000	iARC	3	3	14	12
		iR_Newton	4 (%75)	3 (%67)	9	4
BOX3	3	iARC	7	7	30	32
		iR_Newton	7 (%100)	7 (%100)	16	0
BOXPOWER	20000	iARC	3	3	10	9
		iR_Newton	7 (%100)	7 (%100)	13	0
BRKMCC	2	iARC	2	2	6	7
		iR_Newton	2 (%100)	2 (%100)	4	0
BROWNAL	200	iARC	2	2	6	4
		iR_Newton	1 (%100)	1 (%100)	1	0
BROWNBS	2	iARC	53	38	142	191
		iR_Newton	5 (%80)	5 (%80)	11	5
BROWNDEN	4	iARC	8	8	35	35
		iR_Newton	9 (%100)	9 (%100)	20	0
BROYDN7D	5000	iARC	472	279	7202	12598
		iR_Newton	812 (%29)	346 (%2)	5033	10022
BRYBND	5000	iARC	19	10	240	367
		iR_Newton	17 (%53)	11 (%82)	206	81
CHAINWOO	4000	iARC	81	57	798	942
		iR_Newton	70 (%81)	59 (%88)	409	185
CHNROSNB	50	iARC	64	40	1126	1456
		iR_Newton	53 (%75)	40 (%82)	499	294
CHNRSNBM	50	iARC	96	58	1708	2320
		iR_Newton	101 (%58)	59 (%59)	899	960
CLIFF	2	iARC	14	14	28	14
		iR_Newton	14 (%100)	14 (%100)	14	0
COSINE	10000	iARC	12	7	108	140
		iR_Newton	11 (%55)	7 (%71)	45	29
CRAGGLVY	5000	iARC	30	30	228	208
		iR_Newton	31 (%100)	31 (%100)	108	0
CUBE	2	iARC	42	27	126	170
		iR_Newton	35 (%69)	25 (%76)	72	51
CURLY10	10000	iARC	---	---	---	---

Table B.1 -- continued from previous page

Prob	#Var	Alg	#Iter (%Newton)	#Acc (%Newton)	#Hv-prod	#T-fact
		iR_Newton	328 (%95)	318 (%97)	271881	19957
CURLY30	10000	iARC	---	---	---	---
		iR_Newton	87 (%83)	77 (%91)	125639	630
DENSCHNA	2	iARC	5	5	15	15
		iR_Newton	5 (%100)	5 (%100)	10	0
DENSCHNB	2	iARC	7	6	20	22
		iR_Newton	7 (%71)	5 (%80)	12	8
DENSCHNC	2	iARC	13	9	38	46
		iR_Newton	11 (%82)	9 (%89)	21	8
DENSCHND	3	iARC	61	57	206	172
		iR_Newton	44 (%86)	40 (%95)	82	27
DENSCHNE	3	iARC	24	15	68	62
		iR_Newton	21 (%52)	16 (%69)	41	58
DENSCHNF	2	iARC	5	5	15	15
		iR_Newton	5 (%100)	5 (%100)	10	0
DIXMAANA	3000	iARC	6	6	14	8
		iR_Newton	6 (%100)	6 (%100)	7	0
DIXMAANB	3000	iARC	7	7	16	9
		iR_Newton	7 (%100)	7 (%100)	8	0
DIXMAANC	3000	iARC	8	8	18	9
		iR_Newton	8 (%100)	8 (%100)	9	0
DIXMAAND	3000	iARC	9	9	20	10
		iR_Newton	9 (%100)	9 (%100)	10	0
DIXMAANE	3000	iARC	59	59	670	622
		iR_Newton	60 (%100)	60 (%100)	331	0
DIXMAANF	3000	iARC	38	37	510	487
		iR_Newton	37 (%100)	37 (%100)	249	0
DIXMAANG	3000	iARC	39	39	532	514
		iR_Newton	40 (%100)	40 (%100)	288	0
DIXMAANH	3000	iARC	41	41	448	421
		iR_Newton	41 (%100)	41 (%100)	224	0
DIXMAANI	3000	iARC	193	193	3456	3464
		iR_Newton	249 (%100)	249 (%100)	2814	0
DIXMAANJ	3000	iARC	34	34	324	296
		iR_Newton	34 (%100)	34 (%100)	162	0
DIXMAANK	3000	iARC	30	30	248	225
		iR_Newton	30 (%100)	30 (%100)	124	0
DIXMAANL	3000	iARC	29	29	180	148
		iR_Newton	29 (%100)	29 (%100)	90	0
DIXMAANM	3000	iARC	375	375	10902	11542
		iR_Newton	398 (%100)	398 (%100)	6126	0
DIXMAANN	3000	iARC	82	82	1368	1358
		iR_Newton	87 (%100)	87 (%100)	789	0
DIXMAANO	3000	iARC	63	63	908	893
		iR_Newton	59 (%100)	59 (%100)	371	0
DIXMAANP	3000	iARC	51	51	476	432
		iR_Newton	51 (%100)	51 (%100)	238	0
DIXON3DQ	10000	iARC	2257	2256	143968	164858
		iR_Newton	2476 (%100)	2476 (%100)	81042	0
DJTL	2	iARC	215	120	642	866
		iR_Newton	204 (%32)	81 (%5)	404	512
DQDRTIC	5000	iARC	6	6	34	32
		iR_Newton	4 (%100)	4 (%100)	10	0
DQRTIC	5000	iARC	15	15	30	15
		iR_Newton	11 (%100)	11 (%100)	11	0
EDENSCH	2000	iARC	15	15	44	25
		iR_Newton	15 (%100)	15 (%100)	22	0

Table B.1 -- continued from previous page

Prob	#Var	Alg	#Iter (%Newton)	#Acc (%Newton)	#Hv-prod	#T-fact
EG2	1000	iARC	3	3	6	3
		iR_Newton	3 (%100)	3 (%100)	3	0
EIGENALS	2550	iARC	179	134	15548	20388
		iR_Newton	173 (%84)	150 (%89)	7871	1999
ENGVAL1	5000	iARC	9	9	64	54
		iR_Newton	9 (%100)	9 (%100)	32	0
ENGVAL2	3	iARC	21	15	100	139
		iR_Newton	21 (%57)	15 (%80)	56	34
ERRINROS	50	iARC	131	122	1202	1106
		iR_Newton	108 (%94)	103 (%97)	504	37
ERRINRSM	50	iARC	404	396	6566	7225
		iR_Newton	167 (%98)	163 (%99)	1154	27
EXPFIT	2	iARC	14	9	42	62
		iR_Newton	11 (%27)	6 (%50)	26	38
EXTROSNB	1000	iARC	179	107	2978	3586
		iR_Newton	185 (%62)	114 (%64)	1576	1553
FLETBV3M	5000	iARC	41	34	86	43
		iR_Newton	56 (%43)	32 (%41)	65	32
FLETCHCR	1000	iARC	2437	1450	66056	90373
		iR_Newton	2187 (%66)	1438 (%69)	29012	23819
FMINSRF2	5625	iARC	875	567	6528	7378
		iR_Newton	905 (%50)	448 (%40)	2666	1989
FREUROTH	5000	iARC	17	11	102	120
		iR_Newton	18 (%39)	10 (%60)	51	35
GENHUMPS	5000	iARC	14931	11710	477824	1724919
		iR_Newton	3567 (%2)	2077 (%1)	25952	85744
GENROSE	500	iARC	593	350	24494	54811
		iR_Newton	690 (%19)	341 (%4)	11862	30583
GROWTHLS	3	iARC	8	8	32	32
		iR_Newton	8 (%100)	8 (%100)	16	0
GULF	3	iARC	46	31	196	249
		iR_Newton	40 (%62)	29 (%62)	101	78
HAIRY	2	iARC	28	15	84	133
		iR_Newton	19 (%21)	10 (%40)	45	64
HATFLDD	3	iARC	23	19	108	140
		iR_Newton	23 (%65)	18 (%78)	64	33
HATFLDE	3	iARC	23	18	112	153
		iR_Newton	24 (%62)	18 (%83)	69	45
HATFLDFL	3	iARC	851	711	4255	5134
		iR_Newton	1127 (%84)	961 (%85)	3429	1313
HEART6LS	6	iARC	1502	895	15002	27063
		iR_Newton	1071 (%52)	620 (%51)	5164	6637
HEART8LS	8	iARC	115	69	1407	2400
		iR_Newton	186 (%31)	97 (%23)	1178	2185
HELIX	3	iARC	11	7	52	72
		iR_Newton	15 (%33)	9 (%44)	48	57
HILBERTA	2	iARC	5	5	12	9
		iR_Newton	3 (%100)	3 (%100)	4	0
HILBERTB	10	iARC	4	4	16	12
		iR_Newton	3 (%100)	3 (%100)	6	0
HIMMELBB	2	iARC	10	6	26	36
		iR_Newton	11 (%27)	6 (%33)	20	27
HIMMELBF	4	iARC	53	36	310	408
		iR_Newton	70 (%71)	62 (%77)	244	185
HIMMELBG	2	iARC	7	6	21	25
		iR_Newton	7 (%57)	6 (%67)	13	3
HIMMELBH	2	iARC	5	4	13	12

Table B.1 -- continued from previous page

Prob	#Var	Alg	#Iter	(%Newton)	#Acc	(%Newton)	#Hv-prod	#T-fact
		iR_Newton	6	(%67)	4	(%75)	7	2
HUMPS	2	iARC	125		80		370	655
		iR_Newton	89	(%13)	49	(%10)	218	291
HYDC20LS	99	iARC	11		9		402	539
		iR_Newton	11	(%73)	9	(%89)	215	165
JENSMP	2	iARC	8		8		24	26
		iR_Newton	8	(%100)	8	(%100)	16	0
JIMACK	3549	iARC	54		54		36564	45769
		iR_Newton	52	(%100)	52	(%100)	16267	0
KOWOSB	4	iARC	20		20		114	121
		iR_Newton	18	(%94)	18	(%94)	55	4
LIARWHD	5000	iARC	12		12		46	45
		iR_Newton	11	(%100)	11	(%100)	21	0
LOGHAIRY	2	iARC	167		116		390	414
		iR_Newton	326	(%39)	233	(%49)	542	460
MANCINO	100	iARC	6		6		14	8
		iR_Newton	4	(%100)	4	(%100)	5	0
MARATOSB	2	iARC	3		3		7	5
		iR_Newton	3	(%100)	3	(%100)	4	0
MEXHAT	2	iARC	11		11		33	43
		iR_Newton	11	(%100)	11	(%100)	22	0
MEYER3	3	iARC	12		12		48	51
		iR_Newton	16	(%75)	15	(%73)	38	20
MODBEALE	20000	iARC	---		---		---	---
		iR_Newton	3317	(%99)	3304	(%100)	65293	351
MOREBV	5000	iARC	4		4		1102	2064
		iR_Newton	1	(%100)	1	(%100)	401	0
MSQRTALS	1024	iARC	39		33		9830	12602
		iR_Newton	44	(%73)	36	(%83)	4743	149
MSQRTBLS	1024	iARC	32		26		5822	7090
		iR_Newton	39	(%69)	31	(%81)	3131	156
NCB20	5010	iARC	106		70		2888	4664
		iR_Newton	65	(%32)	43	(%42)	688	614
NCB20B	5000	iARC	29		18		4286	9958
		iR_Newton	38	(%47)	19	(%42)	3297	8386
NONCVXU2	5000	iARC	10302		10302		20604	10302
		iR_Newton	11094	(%100)	11094	(%100)	11094	0
NONCVXUN	5000	iARC	23771		23771		47542	23771
		iR_Newton	20913	(%100)	20913	(%100)	20913	0
NONDIA	5000	iARC	2		2		4	2
		iR_Newton	2	(%100)	2	(%100)	2	0
NONDQUAR	5000	iARC	45		37		156	126
		iR_Newton	38	(%95)	36	(%97)	70	2
OSBORNEA	5	iARC	36		28		289	412
		iR_Newton	21	(%43)	12	(%67)	75	73
OSBORNEB	11	iARC	25		19		396	539
		iR_Newton	28	(%75)	23	(%78)	225	105
OSCIGRAD	100000	iARC	13		10		190	220
		iR_Newton	15	(%40)	9	(%56)	92	61
OSCIPATH	10	iARC	222974		131997		4233806	5617938
		iR_Newton	227426	(%59)	134306	(%59)	2273151	2521354
PALMER1C	8	iARC	161		161		482	362
		iR_Newton	74	(%100)	74	(%100)	145	0
PALMER1D	7	iARC	1069		1069		3586	2567
		iR_Newton	196	(%100)	196	(%100)	379	0
PALMER2C	8	iARC	109		109		326	245
		iR_Newton	76	(%100)	76	(%100)	147	0

Table B.1 -- continued from previous page

Prob	#Var	Alg	#Iter (%Newton)	#Acc (%Newton)	#Hv-prod	#T-fact
PALMER3C	8	iARC	64	64	252	201
		iR_Newton	36 (%100)	36 (%100)	69	0
PALMER4C	8	iARC	27	27	102	84
		iR_Newton	90 (%100)	90 (%100)	177	0
PALMER5C	6	iARC	9	9	22	13
		iR_Newton	10 (%100)	10 (%100)	13	0
PALMER6C	8	iARC	238	238	870	654
		iR_Newton	252 (%100)	252 (%100)	503	0
PALMER7C	8	iARC	65	65	196	143
		iR_Newton	65 (%100)	65 (%100)	120	0
PALMER8C	8	iARC	76	76	300	229
		iR_Newton	90 (%100)	90 (%100)	174	0
PARKCH	15	iARC	31	22	478	685
		iR_Newton	33 (%61)	23 (%74)	270	209
PENALTY1	1000	iARC	14	14	28	14
		iR_Newton	12 (%100)	12 (%100)	12	0
PENALTY2	200	iARC	22	22	314	319
		iR_Newton	22 (%100)	22 (%100)	157	0
PENALTY3	200	iARC	24	20	168	161
		iR_Newton	25 (%60)	18 (%72)	90	44
POWELLSG	5000	iARC	17	17	98	91
		iR_Newton	17 (%100)	17 (%100)	49	0
QUARTC	5000	iARC	15	15	30	15
		iR_Newton	11 (%100)	11 (%100)	11	0
ROSENBR	2	iARC	29	20	87	116
		iR_Newton	32 (%62)	20 (%70)	64	51
S308	2	iARC	12	9	36	41
		iR_Newton	10 (%80)	8 (%88)	20	8
SCHMVETT	5000	iARC	5	5	142	166
		iR_Newton	6 (%100)	6 (%100)	89	0
SENSORS	100	iARC	17	12	146	263
		iR_Newton	21 (%19)	12 (%33)	66	85
SINEVAL	2	iARC	66	42	194	256
		iR_Newton	63 (%63)	41 (%68)	123	81
SINQUAD	5000	iARC	16	11	64	63
		iR_Newton	15 (%33)	9 (%44)	32	26
SISSER	2	iARC	12	12	24	12
		iR_Newton	12 (%100)	12 (%100)	12	0
SNAIL	2	iARC	103	63	290	364
		iR_Newton	107 (%55)	63 (%56)	203	182
SPARSINE	5000	iARC	153	143	15246	18485
		iR_Newton	188 (%88)	174 (%94)	10745	183
SPARSQUR	10000	iARC	15	15	64	49
		iR_Newton	15 (%100)	15 (%100)	32	0
SPMSRTLS	4999	iARC	17	15	582	761
		iR_Newton	17 (%76)	15 (%87)	275	4
SROSENBR	5000	iARC	9	7	36	35
		iR_Newton	10 (%70)	7 (%86)	20	12
SSBRYBND	5000	iARC	75	45	77075	177454
		iR_Newton	39 (%38)	23 (%52)	22010	11269
STRATEC	10	iARC	74	65	886	1069
		iR_Newton	67 (%90)	61 (%95)	413	87
TESTQUAD	5000	iARC	162	162	16908	19812
		iR_Newton	163 (%100)	163 (%100)	8271	0
TOINTGOR	50	iARC	11	11	234	275
		iR_Newton	11 (%100)	11 (%100)	117	0
TOINTGSS	5000	iARC	4	4	14	10

Table B.1 -- continued from previous page

Prob	#Var	Alg	#Iter (%Newton)	#Acc (%Newton)	#Hv-prod	#T-fact
		iR_Newton	3 (%100)	3 (%100)	7	0
TOINTPSP	50	iARC	35	22	254	335
		iR_Newton	41 (%49)	20 (%40)	156	66
TOINTQOR	50	iARC	7	7	104	103
		iR_Newton	7 (%100)	7 (%100)	52	0
TQUARTIC	5000	iARC	11	11	44	50
		iR_Newton	1 (%100)	1 (%100)	2	0
TRIDIA	5000	iARC	16	16	2128	2630
		iR_Newton	17 (%100)	17 (%100)	1310	0
VARDIM	200	iARC	12	12	24	12
		iR_Newton	12 (%100)	12 (%100)	12	0
VAREIGVL	50	iARC	5	5	42	38
		iR_Newton	5 (%100)	5 (%100)	21	0
VIBRBEAM	8	iARC	70	41	644	1131
		iR_Newton	39 (%31)	25 (%48)	183	226
WATSON	12	iARC	14	14	174	214
		iR_Newton	14 (%100)	14 (%100)	88	0
WOODS	4000	iARC	15	15	40	26
		iR_Newton	172 (%87)	157 (%92)	404	144
YFITU	3	iARC	54	38	270	348
		iR_Newton	55 (%65)	39 (%64)	172	130
ZANGWIL2	2	iARC	3	3	6	3
		iR_Newton	1 (%100)	1 (%100)	1	0

Biography

Mohammadreza Samadi, Ph.D., joined the Department of Industrial & Systems Engineering at Lehigh University in Fall 2013. He received his Bachelors and Masters degrees in Industrial Engineering from Sharif University of Technology, Tehran, Iran. His research revolves around designing efficient algorithms for nonconvex optimization problems. His research has been published in top-tier journals in the field of optimization such as *Mathematical Programming*, the *SIAM Journal on Optimization* and the *IMA Journal of Numerical Analysis*.

Education

- Sep 2013–Jan 2019, Lehigh University, Ph.D., Bethlehem, PA, USA.
- Sep 2009–Jan 2012, Sharif University of Technology, M.Sc., Tehran, Iran.
- Sep 2005–Jun 2009, Sharif University of Technology, B.Sc., Tehran, Iran.

Publications

- 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. *Mathematical Programming*, 162(1):1–32, 2017.
- F. E. Curtis, D. P. Robinson, and M. Samadi. Complexity Analysis of a Trust Funnel Algorithm for Equality Constrained Optimization. *SIAM Journal on Optimization*, 28(2), 1533–1563, 2018.

- 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. Accepted for publication in the *IMA Journal of Numerical Analysis*, 2018. doi: 10.1093/imanum/dry022. URL http://dx.doi.org/10.1093/imanum/dry022.

Awards and Fellowships

- Operations Research Summer Fellowship, SAS Institute, 2018.
- INFORMS Magna Cum Laude award for Lehigh University INFORMS Student Chapter, 2015.
- Rossin Doctoral Fellowship, Lehigh University, 2015.
- Dean's Doctoral Student Assistantship, Lehigh University, 2013.
- Exceptionally Talented Student Full Scholarship for Graduate Studies, Sharif University of Technology, 2009.
- Exceptionally Talented Student Full Scholarship for Graduate Studies, Amirkabir University of Technology, 2009.
- Full Scholarship for Undergraduate Studies, Sharif University of Technology, 2005.
- Ranked in the top %1, National University Entrance Examination, 2005.