ARTICLE IN PRESS

Journal of Complexity ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Journal of Complexity

journal homepage: www.elsevier.com/locate/jco

Complexity bounds for second-order optimality in unconstrained optimization

C. Cartis [a,*], N.I.M. Gould [b], Ph.L. Toint [c]

[a] School of Mathematics, University of Edinburgh, The King's Buildings, Edinburgh, EH9 3JZ, Scotland, UK

[b] Computational Science and Engineering Department, Rutherford Appleton Laboratory, Chilton, Oxfordshire, OX11 0QX, England, UK

[c] Namur Center for Complex Systems (naXys), FUNDP—University of Namur, 61, rue de Bruxelles, B-5000 Namur, Belgium

ARTICLE INFO

Article history:
Received 17 December 2010
Accepted 26 May 2011
Available online xxxx

Keywords:
Evaluation complexity
Worst-case analysis
Nonconvex optimization
Second-order optimality conditions

ABSTRACT

This paper examines worst-case evaluation bounds for finding weak minimizers in unconstrained optimization. For the cubic regularization algorithm, Nesterov and Polyak (2006) [15] and Cartis et al. (2010) [3] show that at most $O(\epsilon^{-3})$ iterations may have to be performed for finding an iterate which is within ϵ of satisfying second-order optimality conditions. We first show that this bound can be derived for a version of the algorithm, which only uses one-dimensional global optimization of the cubic model and that it is sharp. We next consider the standard trust-region method and show that a bound of the same type may also be derived for this method, and that it is also sharp in some cases. We conclude by showing that a comparison of the bounds on the worst-case behaviour of the cubic regularization and trust-region algorithms favours the first of these methods.

1. Introduction

We consider algorithms for the solution of the unconstrained (possibly nonconvex) optimization problem

$$\min_x f(x) \tag{1.1}$$

where we assume that $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is smooth (in a sense to be specified later) and bounded below. All methods for the solution of (1.1) are iterative and, starting from some initial guess x_0, generate a sequence $\{x_k\}$ of iterates approximating a critical point of f. Many such algorithms exist, and they are

* Corresponding author.
E-mail addresses: coralia.cartis@ed.ac.uk (C. Cartis), nick.gould@sftc.ac.uk (N.I.M. Gould), philippe.toint@fundp.ac.be (Ph.L. Toint).

0885-064X/$ – see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.jco.2011.06.001

often classified according to their requirements in terms of computing derivatives of the objective function. In this paper, we focus on second-order methods, that is, methods which evaluate the objective function $f(x)$, its gradient $g(x)$ and its Hessian $H(x)$ (or an approximation thereof) at every iteration. The advantage of these methods is that they can be expected to converge to solutions x_* satisfying the second-order optimality conditions

$$\nabla_x f(x_*) = 0, \quad \text{and} \quad \lambda_{\min}(H(x_*)) \geq 0 \tag{1.2}$$

where $\lambda_{\min}(A)$ is the smallest eigenvalue of the symmetric matrix A, rather than only satisfying first-order optimality (i.e., the first of these relations). In practice, however, a second-order algorithm is typically terminated as soon as an iterate x_k is found which is within ϵ of satisfying (1.2), that is, such that

$$\|\nabla_x f(x_k)\| \leq \epsilon_g \quad \text{and} \quad \lambda_{\min}(H(x_k)) \geq -\epsilon_H, \tag{1.3}$$

for some user-specified tolerances $\epsilon_g, \epsilon_H \in (0, 1)$, where $\|\cdot\|$ denotes the Euclidean norm. It is then of interest to bound the number of iterations which may be necessary to find an iterate satisfying (1.3) as a function of the thresholds ϵ_g and ϵ_H. It is the purpose of worst-case complexity analysis to derive such bounds. Many results are available in the literature for the case where the objective function f is convex (see, for instance, [13,14,12,1]). The convergence to approximate first-order points in the nonconvex case has also been investigated for some time (see [16–18,15,10,3–5,8], or [19]).

Of particular interest here is the Adaptive Regularization with Cubics (ARC) algorithm independently proposed by Griewank [11], Weiser et al. [20] and Nesterov and Polyak [15], whose worst-case complexity was shown in the last of these references to be of $O(\epsilon_g^{-3/2})$ iterations for finding an iterate x_k satisfying the approximate first-order optimality conditions (the first relation in (1.3) only) and of $O(\epsilon_H^{-3})$ iterations for finding an iterate x_k satisfying the whole of (1.3).[1] These results were extended by Cartis et al. [3] to an algorithm no longer requiring the computation of exact second-derivatives (but merely of a suitably accurate approximation), nor an (also possibly approximate) knowledge of the objective function's Hessian's Lipschitz constant. More importantly, these authors showed that the $O(\epsilon_g^{-3/2})$ complexity bound for convergence to first-order critical points can be achieved without requiring multi-dimensional global optimization of the cubic model (see [6]). However, such a global minimization on nested Krylov subspaces of increasing dimensions was still required to obtain the $O(\epsilon_H^{-3})$ convergence to second-order critical points.

The present paper focuses on worst-case complexity bounds for convergence to second-order critical points and shows that, as in the first-order case, multi-dimensional global minimization of the cubic model is unnecessary for obtaining the mentioned $O(\epsilon_H^{-3})$ bound for the ARC algorithm. This latter bound is also shown to be sharp. We also prove that a bound of the same type holds for the standard trust-region method. Moreover, we show that it is also sharp for a range of relative values of ϵ_g and ϵ_H. We finally compare the known bounds for the ARC and trust-region algorithms and show that the ARC algorithm is always as good or better from this point of view.

The ARC algorithm is recalled in Section 2 and the associated complexity bounds are derived without multi-dimensional global minimization. Section 3 then discusses an example showing that the bound on convergence of the ARC algorithm to approximate second-order critical points is sharp. A bound of this type is derived in Section 4 for the trust-region methods, its sharpness for suitable values of ϵ_g and ϵ_H is demonstrated, and the comparison with the ARC algorithm discussed. Conclusions and perspectives are finally presented in Section 5.

2. The ARC algorithm and its worst-case complexity

The Adaptive Regularization with Cubics (ARC) algorithm is based on the approximate minimization, at iteration k, of the (possibly nonconvex) cubic model

$$m_k(s) = \langle g_k, s\rangle + \frac{1}{2}\langle s, B_k s\rangle + \frac{1}{3}\sigma_k\|s\|^3, \tag{2.1}$$

[1] It appears that this latter result is the first worst-case complexity bound for convergence to approximate second-order critical points ever proved.

were $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product. Here B_k is a symmetric $n \times n$ approximation of $H(x_k) \overset{\text{def}}{=} H_k$, $\sigma_k > 0$ is a regularization weight and $g_k = \nabla_x m_k(0) = \nabla_x f(x_k)$. By "approximate minimization", we mean that a step s_k is computed to ensure the following conditions.

We first require that the step satisfies the conditions

$$\langle g_k, s_k \rangle + \langle s_k, B_k s_k \rangle + \sigma_k \|s_k\|^3 = 0 \tag{2.2}$$

and

$$\langle s_k, B_k s_k \rangle + \sigma_k \|s_k\|^3 \geq 0. \tag{2.3}$$

As noted in [3], these conditions must hold if s_k is a global minimizer of m_k *along the direction* s_k, that is if $\arg\min_{\alpha \in \mathbb{R}} m_k(\alpha s_k) = 1$ (see Lemma 3.2 in [2]). In order to guarantee convergence to first-order critical points, we also require the familiar "Cauchy condition"

$$m_k(s_k) \leq m_k(s_k^{\rm C}) \tag{2.4}$$

with

$$s_k^{\rm C} = -\alpha_k^{\rm C} g_k \quad \text{and} \quad \alpha_k^{\rm C} = \arg\min_{\alpha \geq 0} m_k(-\alpha g_k). \tag{2.5}$$

Because we are, in addition, interested in convergence to second-order critical points, we also require the following variant of the "eigencondition" whenever B_k is not positive semi-definite (see Section 6.6.1 in [9]): we require in that case that

$$m_k(s_k) \leq m_k(s_k^{\rm E}), \tag{2.6}$$

where

$$s_k^{\rm E} = \alpha_k^{\rm E} u_k \text{ and } \alpha_k^{\rm E} = \arg\min_{\alpha \geq 0} m_k(\alpha u_k), \tag{2.7}$$

with u_k being an approximate eigenvector of B_k associated with its smallest eigenvalue $\lambda_{\min}(B_k) \overset{\text{def}}{=} \tau_k$, in the sense that

$$\langle g_k, u_k \rangle \leq 0 \text{ and } \langle u_k, B_k u_k \rangle \leq \kappa_{\rm snc} \tau_k \|u_k\|^2 \tag{2.8}$$

for some constant $\kappa_{\rm snc} \in (0, 1]$. The knowledge of τ_k and u_k may be obtained, for instance, by applying the (inverse) power method to B_k. Note that we require the minimization in (2.5) and (2.7) to be global, which means that (2.2) and (2.3) also hold with s_k replaced by $s_k^{\rm C}$ and $s_k^{\rm E}$. Finally, we may also optionally require that

$$\|\nabla_x m_k(s_k)\| = \|g_k + B_k s_k + \sigma_k \|s_k\| s_k\| \leq \kappa_\theta \min[1, \|s_k\|] \|g_k\|, \tag{2.9}$$

for some given constant $\kappa_\theta \in (0, 1)$ if we wish to accelerate the convergence to first-order critical points.

Remarkably, conditions (2.2)–(2.9) can all be ensured algorithmically and hold, in particular, if s_k is a global minimizer of m_k (see [11,15], see also [2,7]) which can be computed in polynomial time (see Section 5.1 of [15]). We also note that, if s_k is computed as the global minimizer of m_k in a subspace $\mathcal{L}_k$ containing the gradient and satisfies (2.9), then all the above conditions also hold with $u_k = Q_k w_k$, where τ_k and w_k are respectively the most negative eigenvalue of $Q_k^T B_k Q_k$ and its corresponding eigenvector, and Q_k is an orthonormal basis of $\mathcal{L}_k$. We also note that they require global minimization of the cubic model along $-g_k$, (possibly) u_k and s_k, but that global minimization in subspaces of dimension larger than one is not necessary.

The ARC algorithm may then be stated as presented on the following page. In this description, we assume that the constants satisfy $1 \leq \gamma_1 \leq \gamma_2$, $0 < \eta_1 \leq \eta_2 < 1$ and $\sigma_0 \geq \sigma_{\min} > 0$.

```
Algorithm 2.1: ARC algorithm

Step 0: A starting point $x_0$, an initial and a minimal regularization parameter $\sigma_0 \geq \sigma_{\min}$, and user-
        defined accuracy thresholds $\epsilon_g, \epsilon_H \in (0, 1)$ are given. Set $k = 0$.
Step 1: If conditions (1.3) hold, terminate with approximate solution $x_k$.
Step 2: Compute a Hessian approximation $B_k$ and a step $s_k$ satisfying (2.2)–(2.9).
Step 3: Compute $f(x_k + s_k)$ and
            $\rho_k = \frac{f(x_k) - f(x_k + s_k)}{-m_k(s_k)}$.                                        (2.10)
        Set $x_{k+1} = x_k + s_k$ if $\rho_k \geq \eta_1$, or $x_{k+1} = x_k$ otherwise.
Step 4: Set
            $\sigma_{k+1} \in [\sigma_{\min}, \sigma_k]$            if $\rho_k > \eta_2$,                  [very successful iteration]
            $\sigma_{k+1} \in [\sigma_k, \gamma_1\sigma_k]$         if $\eta_1 \leq \rho_k \leq \eta_2$,   [successful iteration]      (2.11)
            $\sigma_{k+1} \in [\gamma_1\sigma_k, \gamma_2\sigma_k]$ otherwise.                             [unsuccessful iteration]
        Increment $k$ by one and return to Step 1.
```

Let $\mathcal{S}$ denote the index set of all successful or very successful iterations in the sense of (2.11), and define

$$\mathcal{S}_j = \{k \in \mathcal{S} \mid k \leq j\} \quad \text{and} \quad \mathcal{U}_j = \{0, \ldots, j\} \setminus \mathcal{S}_j, \tag{2.12}$$

the sets of successful and unsuccessful iterations up to iteration j.

We now recall the main complexity results for this method, as well as the assumptions under which these hold. We first restate our assumptions.

A.1: The objective function f is twice continuously differentiable on $\mathbb{R}^n$ and its gradient and Hessian are Lipschitz continuous on the path of iterates with Lipschitz constants L_g and L_H, i.e., for all $k \geq 0$ and all $\alpha \in [0, 1]$,

$$\|\nabla_x f(x_k) - \nabla_x f(x_k + \alpha s_k)\| \leq L_g \alpha \|s_k\| \tag{2.13}$$

and

$$\|\nabla_{xx} f(x_k) - \nabla_{xx} f(x_k + \alpha s_k)\| \leq L_H \alpha \|s_k\|. \tag{2.14}$$

A.2: The objective function f is bounded below, i.e. there exists a constant f_{low} such that, for all $x \in \mathbb{R}^n$,

$$f(x) \geq f_{\text{low}}.$$

A.3: For all $k \geq 0$, the Hessian approximation B_k satisfies

$$\|B_k\| \leq \kappa_{\text{B}} \tag{2.15}$$

and

$$\|(\nabla_{xx} f(x_k) - B_k)s_k\| \leq \kappa_{\text{BH}} \|s_k\|^2 \tag{2.16}$$

for some constants $\kappa_{\text{B}} > 1$ and $\kappa_{\text{BH}} > 0$.

The complexity analysis presented below involves a number of constants which depend on the problem formulation (but not on its dimension) and on the algorithm. More specifically, these constants depend on (a subset of) the Lipschitz constants L_g and L_H, the distance between the initial objective value and a lower bound on the global optimum $f(x_0) - f_{\text{low}}$, the uniform upper bound κ_{B} and κ_{BH} on the approximate Hessians B_k and on the difference $\|H_k - B_K\|$ respectively, and the user-chosen algorithm parameters, namely $\gamma_1, \gamma_2, \eta_1, \eta_2, \sigma_0, \sigma_{\min}, \kappa_{\text{snc}}$ and κ_θ, but we refrain from giving the specific sublist for each considered constant for improved readability.

We start by noting that the form of the cubic model (2.1) and (2.2)–(2.3) ensure a remarkable bound on the step norm and model decrease.

Lemma 2.1 (*Lemma 4.2 in [3]*). *We have that*

$$m_k(s_k) \le -\frac{1}{6}\sigma_k \|s_k\|^3. \tag{2.17}$$

Note that, since (2.2) and (2.3) hold with s_k replaced by s_k^{C} and s_k^{E} and mentioned above, (2.17) thus also holds with s_k replaced by s_k^{C} and s_k^{E}. For our purposes it is also useful to consider the following bounds on the value of the regularization parameter.

Lemma 2.2. *Suppose that (2.13) and (2.15) hold. Then there exists a constant $\kappa_\sigma > 0$ such that, for all $k \ge 0$*

$$\sigma_k \le \max\left[\sigma_0, \frac{\kappa_\sigma}{\epsilon_g}\right]. \tag{2.18}$$

If, in addition, (2.14) and (2.16) also hold, then there exists a constant $\sigma_{\max} > 0$ independent of ϵ_g and ϵ_H such that, for all $k \ge 0$,

$$\sigma_k \le \sigma_{\max}. \tag{2.19}$$

Proof. See Lemmas 3.2 and 3.3 in [3] for the proof of (2.18) and Lemma 5.2 in [2] for that of (2.19). □

A first complexity bound can then be derived.

Lemma 2.3 (*Corollary 3.4 in [3]*). *Assume that (2.13),* A.2 *and (2.15) hold. Then there exists a constant $\kappa^0_{\mathrm{ARC,S}} > 0$ such that $N^1_{\mathrm{ARC,S}}$, the total number of successful and very successful iterations of the ARC algorithm with $\|g_k\| \ge \epsilon_g$, is bounded above by $\lceil \kappa^0_{\mathrm{ARC,S}}\epsilon_g^{-2}\rceil$.*

If we are ready to strengthen our assumption by assuming (2.14) and to impose (2.9), then, crucially, the step s_k can then be proved to be sufficiently long compared to the gradient's norm at iteration $k+1$.

Lemma 2.4 (*Lemma 5.2 in [3]*). *Suppose that* A.1, A.3 *and (2.9) hold. Then, for all $k \ge 0$, one has that, for some $\kappa_g > 0$,*

$$\|s_k\| \ge \kappa_g\sqrt{\|\nabla_x f(x_k + s_k)\|}. \tag{2.20}$$

Combining (2.17) with this last result, it is then not difficult to show the second complexity result.

Lemma 2.5 (*Corollary 5.3 in [3]*). *Suppose that* A.1–A.3 *and (2.9) hold. Then there exists a constant $\kappa^1_{\mathrm{ARC,S}} > 0$ such that $N^1_{\mathrm{ARC,S}}$ (as defined in Lemma 2.3) is bounded above by $\lceil \kappa^1_{\mathrm{ARC,S}}\epsilon_g^{-3/2}\rceil$.*

The final important observation in the first-order analysis is that the total number of iterations required by the ARC algorithm to terminate may be bounded in terms of the number of successful iterations needed.

Lemma 2.6 (*Theorem 2.1 in [3]*). *For any fixed $j \ge 0$, let $\mathcal{S}_j$ and $\mathcal{U}_j$ be defined by (2.12). Then one has that*

$$|\mathcal{U}_j| \le \left\lceil (|\mathcal{S}_j| + 1)\frac{1}{\log \gamma_1}\log\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)\right\rceil. \tag{2.21}$$

We may now use this last result with Lemmas 2.3 and 2.5, and deduce the following worst-case bounds.

Theorem 2.7 (*See Corollary 5.5 in [3]*). *Suppose that (2.13), A.2 and (2.15) hold. Then, the ARC algorithm produces an iterate x_k satisfying the first part of (1.3) after at most*

$$\lceil \kappa^{\mathrm{1st}}_{\mathrm{ARC,S}}\epsilon_g^{-2}\rceil \tag{2.22}$$

successful iterations and at most

$$\lceil \kappa^{\rm 1st}_{\rm ARC} \epsilon_g^{-2} \rceil \tag{2.23}$$

iterations in total, where $\kappa^{\rm 1st}_{\rm ARC,S}$ *and* $\kappa^{\rm 1st}_{\rm ARC}$ *are positive constants. Moreover, if* (2.9) *and* (2.14) *also hold, then the bounds* (2.22) *and* (2.23) *respectively become*

$$\lceil \kappa^{\rm 1st}_{\rm ARC,S} \epsilon_g^{-3/2} \rceil \quad \text{and} \quad \lceil \kappa^{\rm 1st}_{\rm ARC} \epsilon_g^{-3/2} \rceil. \tag{2.24}$$

The bounds (2.24) are known to be qualitatively[2] tight and optimal for a wide class of second-order methods (see [4,5]).

After reviewing the complexity of convergence to first-order critical points, we now turn to the analysis of the number of iterations necessary to ensure the second part of (1.3) under our present assumptions (which do not require multi-dimensional global model minimization).

Lemma 2.8. *Suppose that* A.1–A.3 *hold. Then there exists a constant* $\kappa^2_{\rm ARC,S} > 0$ *such that* $N^2_{\rm ARC,S}$*, the total number of successful and very successful iterations of the ARC algorithm with* $\tau_k < -\epsilon_H$*, is bounded above by* $\lceil \kappa^2_{\rm ARC,S} \epsilon_H^{-3} \rceil$.

Proof. We first note that, when $\tau_k < 0$, $s_k^{\rm E}$ gives a minimizer of the model in the direction u_k by (2.7), from which we derive, using (2.3) for $s_k^{\rm E}$, that, for all $k \geq 0$,

$$\sigma_k \|s_k^{\rm E}\| \geq -\frac{\langle s_k^{\rm E}, B_k s_k^{\rm E}\rangle}{\|s_k^{\rm E}\|^2} \geq \kappa_{\rm snc} |\tau_k|, \tag{2.25}$$

where we have used (2.8) to derive the last inequality. Combining this bound with (2.17) applied for $s_k^{\rm E}$, and (2.19), we then obtain that

$$-m_k(s_k) \geq -m_k(s_k^{\rm E}) \geq \frac{\kappa_{\rm snc}|\tau_k|^3}{6\sigma_k^2} \geq \frac{\kappa_{\rm snc}}{6\sigma_{\max}^2}|\tau_k|^3 \geq \frac{\kappa_{\rm snc}\epsilon_H^3}{6\sigma_{\max}^2} \tag{2.26}$$

for all k such that the second part of (1.3) fails. If we now restrict our attention to the subset of those iterations which are successful or very successful, we obtain, using A.2 and the monotonically decreasing nature of the sequence $\{f(x_k)\}$, that

$$f(x_0) - f_{\rm low} \geq \sum_{k=0, k\in\mathcal{S}} (f(x_k) - f(x_{k+1})) \geq N^2_{\rm ARC,S} \frac{\eta_1 \kappa_{\rm snc} \epsilon_H^3}{6\sigma_{\max}^2}.$$

We therefore obtain the desired result with $\kappa^2_{\rm ARC,S} \overset{\rm def}{=} 6\sigma_{\max}^2 (f(x_0) - f_{\rm low})/\kappa_{\rm snc}\eta_1$. □

As was the case for convergence to first-order critical points, we may now combine Lemmas 2.3 and 2.6 with our last result to obtain worst-case complexity bounds for convergence of the ARC algorithm to approximate second-order critical points.

Theorem 2.9. *Suppose that* A.1–A.3 *hold. Then, the ARC algorithm produces an iterate* x_k *satisfying* (1.3) *(and thus terminates) after at most*

$$\lceil \kappa^{\rm 2nd}_{\rm ARC,S} \max[\epsilon_g^{-2}, \epsilon_H^{-3}] \rceil \tag{2.27}$$

successful or very successful iterations and at most

$$\lceil \kappa^{\rm 2nd}_{\rm ARC} \max[\epsilon_g^{-2}, \epsilon_H^{-3}] \rceil \tag{2.28}$$

iterations in total, where $\kappa^{\rm 2nd}_{\rm ARC,S}$ *and* $\kappa^{\rm 2nd}_{\rm ARC}$ *are positive constants. Moreover, if* (2.9) *also holds, then the bounds* (2.27) *and* (2.28) *respectively become*

[2] The constants may not be optimal.

$$\lceil \kappa^{2nd}_{ARC,S} \max[\epsilon_g^{-3/2}, \epsilon_H^{-3}]\rceil \quad \textit{and} \quad \lceil \kappa^{2nd}_{ARC} \max[\epsilon_g^{-3/2}, \epsilon_H^{-3}]\rceil. \tag{2.29}$$

Proof. Lemmas 2.3 and 2.8 yield that the total number of successful iterations such that the first or the second part of (1.3) is violated cannot exceed

$$\kappa^0_{ARC,S}\epsilon_g^{-2} + \kappa^2_{ARC,S}\epsilon_H^{-3}.$$

We thus immediately deduce (2.27) with $\kappa^{2nd}_{ARC,S} \overset{\text{def}}{=} \kappa^0_{ARC,S} + \kappa^2_{ARC,S}$. The bound (2.28) follows by applying Lemma 2.6, while (2.29) directly is obtained by using Lemma 2.5 instead of Lemma 2.3 in this reasoning. □

3. An example of slow convergence of ARC

We now show by an example that the bounds (2.27) and (2.28) cannot be improved. Our example is unidimensional and is inspired by the technique used in [4,5].

We first choose the starting point and sequences of gradient and Hessian values and steps to be, for all $k \geq 0$,

$$x_0 = 0, \qquad g_k = 0, \qquad s_k = \left(\frac{1}{k+1}\right)^{\frac{1}{3}+\delta} \quad \text{and} \quad B_k = H_k = \tau_k = -\left(\frac{1}{k+1}\right)^{\frac{1}{3}+\delta} \tag{3.1}$$

where $\delta \in (0, 1)$ is a (small) positive constant. Because it is straightforward to verify that the conditions (2.2)–(2.9) hold with this choice and $\sigma_k = 1$ for all k, we may consider these values as produced by the kth iteration of the ARC algorithm at iterate $x_k = x_0 + \sum_{j=0}^{k-1} s_j$. We also define $f_k \overset{\text{def}}{=} f(x_k)$ for all k by the relations

$$f_0 = \zeta(1+3\delta) \quad \text{and} \quad f_{k+1} = f_k - \left(\frac{1}{k+1}\right)^{1+3\delta}, \tag{3.2}$$

where $\zeta(t) \overset{\text{def}}{=} \sum_{k=1}^{\infty} k^{-t}$ is the Riemann zeta function, which is finite for all $t > 1$ (and thus for $t = 1+3\delta$). Observe that, since (2.2) and (2.3) both hold as equalities, we have that

$$-m_k(s_k) = \frac{1}{6}\|s_k\|^3 = \frac{1}{6}\left(\frac{1}{k+1}\right)^{1+3\delta}$$

and (3.2) therefore implies that all iterations are very successful, allowing us to keep σ_k fixed to 1.

We now use Hermite interpolation to construct the objective function f on the successive intervals $[x_k, x_{k+1}]$, and define

$$f(x) = p_k(x - x_k) + f_{k+1} \quad \text{for } x \in [x_k, x_{k+1}] \text{ and } k \geq 0, \tag{3.3}$$

where p_k is the polynomial

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

with coefficients defined by the interpolation conditions

$$\begin{array}{ll} p_k(0) = f_k - f_{k+1}, & p_k(s_k) = 0; \\ p_k'(0) = g_k, & p_k'(s_k) = g_{k+1}; \\ p_k''(0) = H_k, & p_k''(s_k) = H_{k+1}. \end{array} \tag{3.4}$$

These conditions yield the following values for the first three coefficients

$$c_{0,k} = f_k - f_{k+1}, \qquad c_{1,k} = g_k = 0, \qquad c_{2,k} = \frac{1}{2}H_k;$$

and the remaining coefficients satisfy

$$\begin{pmatrix} s_k^3 & s_k^4 & s_k^5 \\ 3s_k^2 & 4s_k^3 & 5s_k^4 \\ 6s_k & 12s_k^2 & 20s_k^3 \end{pmatrix} \begin{pmatrix} c_{3,k} \\ c_{4,k} \\ c_{5,k} \end{pmatrix} = \begin{pmatrix} \Delta f_k - g_k s_k - \frac{1}{2} s_k H_k s_k \\ \Delta g_k - H_k s_k \\ \Delta H_k \end{pmatrix},$$

where

$$\Delta f_k = f_{k+1} - f_k, \qquad \Delta g_k = g_{k+1} - g_k \quad \text{and} \quad \Delta H_k = H_{k+1} - H_k.$$

Hence we obtain, also from (3.1) and $\Delta g_k = 0$, that

$$\begin{aligned} c_{3,k} &= 10\frac{\Delta f_k}{s_k^3} - 4\frac{\Delta g_k}{s_k^2} + \frac{\Delta H_k}{2s_k} - 10\frac{g_k}{s_k^2} - \frac{H_k}{s_k} = 10\frac{\Delta f_k}{s_k^3} + \frac{\Delta H_k}{2s_k} - \frac{H_k}{s_k}; \\ c_{4,k} &= -15\frac{\Delta f_k}{s_k^4} + 7\frac{\Delta g_k}{s_k^3} - \frac{\Delta H_k}{s_k^2} + 15\frac{g_k}{s_k^3} + \frac{H_k}{2s_k^2} = -15\frac{\Delta f_k}{s_k^4} - \frac{\Delta H_k}{s_k^2} + \frac{H_k}{2s_k^2}; \\ c_{5,k} &= 6\frac{\Delta f_k}{s_k^5} - 3\frac{\Delta g_k}{s_k^4} + \frac{\Delta H_k}{2s_k^3} - 6\frac{g_k}{s_k^4} = 6\frac{\Delta f_k}{s_k^5} + \frac{\Delta H_k}{2s_k^3}. \end{aligned} \tag{3.5}$$

It remains to show that the constructed f satisfies A.1–A.3. One easily sees from its construction that f is twice continuously differentiable. Moreover its third derivative exists everywhere and, on the kth interval, satisfies the bound

$$|f'''(x_k + s)| = |p_k'''(s)| \leq 6|c_{3,k}| + 24|c_{4,k}|s_k + 60|c_{5,k}|s_k^2, \quad s \in [0, s_k].$$

But (3.1), (3.5) and the resulting inequality $|\Delta H_k| \leq |H_k|$ imply that $|c_{3,k}|$, $|c_{4,k}|s_k$ and $|c_{5,k}|s_k^2$ are uniformly bounded, and thus so is f'''. Moreover, $|H_k|$, and hence $|c_{2,k}|$, are also bounded, which, combined with the boundness of $|s_k|$, implies that f'' is bounded above. As a consequence, f has Lipschitz continuous gradient and Hessian, and A.1 holds. The definition (3.2) and the definition of the Riemann function together imply that A.2 holds with $f_{\text{low}} = 0$[3] and A.3 directly results from (3.1). Fig. 3.1 shows plots of f and its first three derivatives for $\delta = 0.0001$ and for $k = 0, \ldots, 15$. The figure reveals the objective functions' nonconvexity and monotonically decreasing nature.

We have thus verified that the ARC algorithm applied on f (which satisfies A.1–A.3) starting from $x_0 = 0$ and $\sigma_0 = 1$ produces iterates such that

$$\lambda_{\min}(H_k) = -\left(\frac{1}{k+1}\right)^{\frac{1}{3}+\delta}$$

and for which the second part of (1.3) fails for exactly

$$\left\lceil \frac{1}{\epsilon_H^{\frac{3}{1+3\delta}}} \right\rceil - 1$$

iterations, for any ϵ_H, ϵ_g and δ in $(0, 1)$. Since we know from Cartis et al. [4] that the bound in $O(\epsilon_g^{-3/2})$ is sharp for obtaining a mere first-order approximate critical point, we deduce that the bound (2.29) cannot be improved. As a consequence it is sharp as far as the ARC algorithm is concerned. Finally note that the iterate at termination of the algorithm is finite for any ϵ_g and ϵ_H, and therefore f can be prolongated smoothly beyond that iterate in such a way that it has a unique and finite global minimizer. As a consequence, our reasoning also applies to such minimizers.

[3] Note that we have shown that $f(x)$ is bounded below for $x \geq 0$, which is the domain of interest since $x_k \geq 0$; we may extend f by continuity for $x < 0$.

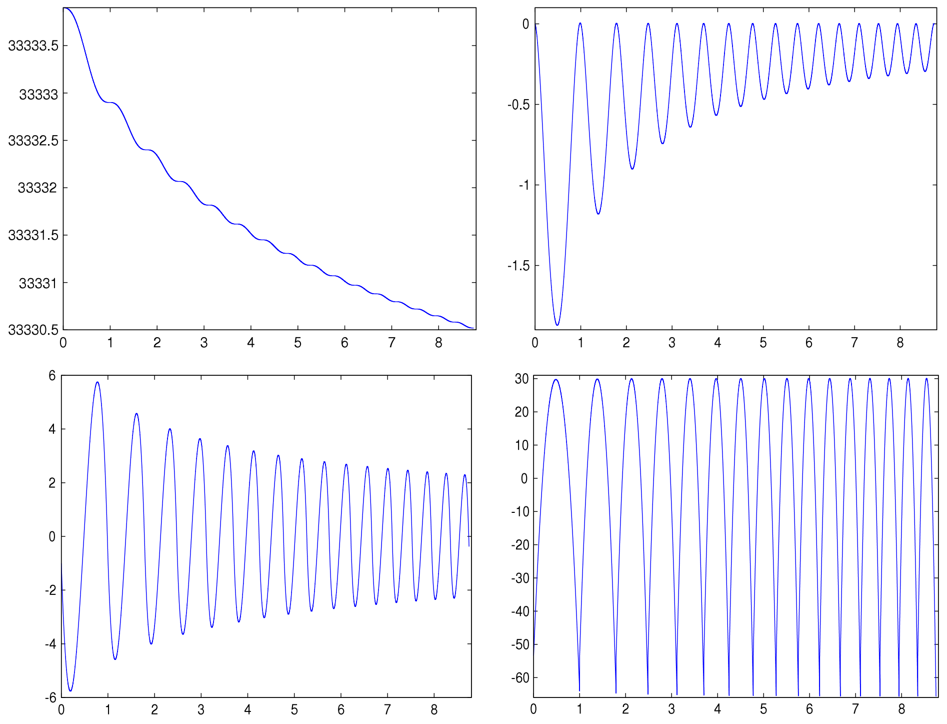

Fig. 3.1. The function f and its first three derivatives (from top to bottom and left to right) on the first 16 intervals.

4. Second-order complexity for the trust-region method

We may wonder if the worst-case complexity for convergence to approximate second-order points is better or worse for the standard trust-region method than for ARC. Our first step is to establish an upper bound on this complexity for the trust-region method, which requires revisiting some of its convergence theory. For the sake of completeness, we briefly recall the basic formulation of this method, as based on Section 6.1 of [9]. The main idea of the trust-region method is similar to that of the ARC algorithm: at iteration k, a quadratic model

$$m_k(s) \overset{\text{def}}{=} \langle g_k, s\rangle + \frac{1}{2}\langle s, B_k s\rangle \tag{4.1}$$

is minimized in the "trust region" defined by

$$\mathcal{B}_k \overset{\text{def}}{=} \{s \in \mathbb{R}^n \mid \|s\| \le \Delta_k\}, \tag{4.2}$$

where Δ_k is the (dynamically updated) trust-region radius. The other conditions on the step s_k are again similar to what happens for the ARC method: one typically requires s_k to satisfy (2.4)–(2.8) where the model $m_k(s)$ is now defined by (4.1) instead of (2.1) and where minimization in (2.4) and (2.6) is restricted to the trust region. Note that, in this context, global optimization of the model along s_k^{C} or s_k^{E} within the trust region no longer implies (2.2) and (2.3). In practice, the condition (2.9) is often replaced by

$$\|\nabla_x m_k(s_k)\| = \|g_k + B_k s_k\| \le \kappa_\theta \min[1, \|g_k\|^\alpha]\|g_k\|, \tag{4.3}$$

for some given constant $\kappa_\theta \in (0, 1)$ and some exponent $\alpha > 0$, but this is irrelevant for the complexity analysis developed below. Global optimization of the model along s_k within the trust region is not necessary.

The basic trust-region algorithm may then be stated as follows.

```
Algorithm 4.1: Trust-region algorithm

Step 0: A starting point $x_0$, an initial radius $\Delta_0 > 0$ and user-defined accuracy thresholds $\epsilon_g, \epsilon_H \in (0, 1)$ are given. Set $k = 0$.
Step 1: If conditions (1.3) hold, terminate with approximate solution $x_k$.
Step 2: Compute a Hessian approximation $B_k$ and a step $s_k \in \mathcal{B}_k$ satisfying (2.4)–(2.8) and (optionally) (4.3).
Step 3: Compute $f(x_k + s_k)$ and $\rho_k$ given by (2.10). Set $x_{k+1} = x_k + s_k$ if $\rho_k \geq \eta_1$, or $x_{k+1} = x_k$ otherwise.
Step 4: Set
        $$\Delta_{k+1} \in \begin{cases} [\Delta_k, \gamma_3\Delta_k]) & \text{if } \rho_k > \eta_2, & \text{[very successful iteration]} \\ [\gamma_2\Delta_k, \Delta_k] & \text{if } \eta_1 \leq \rho_k \leq \eta_2, & \text{[successful iteration]} \\ [\gamma_1\Delta_k, \gamma_2\Delta_k] & \text{otherwise.} & \text{[unsuccessful iteration]} \end{cases} \quad (4.4)$$
        Increment $k$ by one and return to Step 1.
```

In this algorithm, we have assumed that the constants satisfy the inequalities .5

$$\Delta_0 \leq \Delta_{\max}, \quad 0 < \eta_1 \leq \eta_2 < 1 \text{ and } 0 < \gamma_1 \leq \gamma_2 < 1 \leq \gamma_3 \tag{4.5}$$

and we define the sets of very successful, successful and unsuccessful iterations just as in (2.12). As was the case for the analysis of the ARC algorithm, the constants arising in the analysis below will depend on the problem characteristics given by L_g and L_H, the difference $f(x_0) - f_{\text{low}}$, the constants κ_{B} and κ_{BH}, and on the algorithmic parameters, this last set now containing $\Delta_0, \Delta_{\max}, \eta_1, \eta_2, \gamma_1, \gamma_2, \gamma_3, \kappa_{\text{snc}}, \kappa_\theta$ and α.

In order to establish the desired complexity bound, we start by re-examining the size of the discrepancy between the model and the objective function in the case where Lipschitz continuity of the Hessian is assumed (an assumption never made in Chapter 6 of [9]).

Lemma 4.1. *Suppose that* A.1 *and* A.3 *hold. Then, for each $k \geq 0$,*

$$|f(x_k + s_k) - f(x_k) - m_k(s_k)| \leq \kappa_{\text{fm}}\Delta_k^3 \tag{4.6}$$

for some $\kappa_{\text{fm}} > 0$.

Proof. (See the proof of Lemma 6.4.1 in [9].) Using A.1, we may apply the mean-value theorem on the objective function and obtain that

$$f(x_k + s_k) = f(x_k) + \langle g_k, s_k\rangle + \frac{1}{2}\langle s_k, H(\xi_k)s_k\rangle$$

for some ξ_k in the segment $[x_k, x_k + s_k]$. Subtracting (4.1), taking absolute values and using A.1, A.3, the inequality $\|\xi_k - x_k\| \leq \|s_k\|$ and the Cauchy–Schwarz inequality yields that

$$\begin{aligned} |f(x_k + s_k) - f(x_k) - m_k(s_k)| &= \frac{1}{2}|\langle s_k, H(\xi_k)s_k\rangle - \langle s_k, B_k s_k\rangle| \\ &\leq \frac{1}{2}|\langle s_k, [H(\xi_k) - H(x_k) + H(x_k) - B_k]s_k\rangle| \\ &\leq \frac{1}{2}(L_H\|s_k\|^3 + \kappa_{\text{BH}}\|s_k\|^3) \end{aligned}$$

and (4.6) with $\kappa_{\text{fm}} = \frac{1}{2}(L_H + \kappa_{\text{BH}})$ then follows from the inequality $\|s_k\| \leq \Delta_k$. □

We then recall a standard result on the model decrease in the presence of significant gradient or negative curvature.

Lemma 4.2 (*Theorems 6.3.1 and 6.6.1 in [9]*). *Suppose that m_k is given by (4.1). Then, if $\|g_k\| > 0$, we have that*

$$-m_k(s_k) \geq -m_k(s_k^{\rm C}) \geq \frac{1}{2}\|g_k\| \min\left[\frac{\|g_k\|}{\kappa_{\rm B}}, \Delta_k\right] \tag{4.7}$$

while, if $\tau_k < 0$ (where τ_k is given by (2.8)), then

$$-m_k(s_k) \geq -m_k(s_k^{\rm E}) \geq \frac{1}{2}\kappa_{\rm snc}|\tau_k|\Delta_k^2. \tag{4.8}$$

From this result, we may deduce the following crucial lemma.

Lemma 4.3. *Suppose that* A.1 *and* A.2 *hold and that m_k is given by (4.1). Suppose furthermore that $\tau_k < 0$ and that*

$$\Delta_k \leq \frac{(1-\eta_2)\kappa_{\rm snc}|\tau_k|}{2\kappa_{\rm fm}}. \tag{4.9}$$

Then iteration k of the trust-region algorithm is very successful and $\Delta_{k+1} \geq \Delta_k$.

Proof. Suppose that (4.9) holds. We obtain from (4.6) and (4.8) that

$$|\rho_k - 1| = \left|\frac{f(x_k+s_k) - m_k(s_k)}{-m_k(s_k)}\right| \leq \frac{\kappa_{\rm fm}}{\frac{1}{2}\kappa_{\rm snc}|\tau_k|}\Delta_k \leq 1-\eta_2,$$

where we used (4.9) to deduce the last inequality. Thus $\rho_k \geq \eta_2$ and the mechanism of the trust-region algorithm then ensures that iteration k is very successful and, by (4.4), that $\Delta_{k+1} \geq \Delta_k$. □

We may then use this result to show that, as long as second-order optimality is not reached in the sense of (1.3), then the trust-region radius is bounded away from zero. To make our result more precise we first observe that

$$\text{either } \|g_k\| \geq \epsilon_g \quad \text{or} \quad \tau_k \leq -\epsilon_H \tag{4.10}$$

as long as the trust-region algorithm does not terminate.

Lemma 4.4. *Suppose that* A.1 *and* A.2 *hold and that m_k is given by (4.1). Then,there exists a constant $\kappa_\Delta \in (0,1)$ independent of k such that, if the trust-region algorithm does not terminate at iteration k,*

$$\Delta_k \geq \kappa_\Delta \min[\epsilon_g, \epsilon_H]. \tag{4.11}$$

Proof. Assume, for the purpose of deriving a contradiction, that iteration k is the first such that

$$\Delta_{k+1} \leq \gamma_1 \min\left[\frac{1}{2\kappa_{\rm B}}, \frac{\kappa_{\rm snc}}{2\kappa_{\rm fm}}\right](1-\eta_2)\ \min[\epsilon_g, \epsilon_H]. \tag{4.12}$$

Then we have from (4.4) that, either

$$\Delta_k \leq \min\left[\frac{1}{2\kappa_{\rm B}}, \frac{\kappa_{\rm snc}}{2\kappa_{\rm fm}}\right](1-\eta_2)\ \min[\epsilon_g, \epsilon_H] \leq \frac{(1-\eta_2)}{2\kappa_{\rm B}}\epsilon_g \leq \frac{(1-\eta_2)}{2\kappa_{\rm B}}\|g_k\|$$

if the first part of (4.10) holds, or

$$\Delta_k \leq \min\left[\frac{1}{2\kappa_{\rm B}}, \frac{\kappa_{\rm snc}}{2\kappa_{\rm fm}}\right](1-\eta_2)\min[\epsilon_g, \epsilon_H] \leq \frac{(1-\eta_2)\kappa_{\rm snc}}{2\kappa_{\rm fm}}\epsilon_H \leq \frac{(1-\eta_2)\kappa_{\rm snc}|\tau_k|}{2\kappa_{\rm fm}}$$

if the second part of (4.10) holds. In the first case, Theorem 6.4.3 in [9] implies that iteration k is very successful and $\Delta_{k+1} \geq \Delta_k$. In the second case, the same conclusion follows from Lemma 4.3. Thus $\Delta_{k+1} \geq \Delta_k$ in both cases and our assumption that iteration k is the first such that (4.12) holds must be false. As a consequence, there cannot be any iteration such that inequality (4.12) holds as long as the algorithm does not terminate, and we obtain the desired conclusion with

$$\kappa_\Delta \stackrel{\text{def}}{=} \gamma_1 \min\left[\frac{1}{2\kappa_{\rm B}}, \frac{\kappa_{\rm snc}}{2\kappa_{\rm fm}}\right](1-\eta_2) < 1, \tag{4.13}$$

the last inequality following from the bound $\kappa_{\rm B} \geq 1$ and (4.5). □

We may now compute an upper bound on the number of successful or very successful iterations such that (1.3) does not hold.

Lemma 4.5. *Suppose that* A.1 *and* A.2 *hold and that m_k is given by* (4.1)*. Then there exists a constant $\kappa_{\rm TR,S}^{\rm 2nd} > 0$ such that $N_{\rm TR,S}^{\rm 2nd}$, the number of successful or very successful iterations of the trust-region method before* (1.3) *holds, is bounded above by $\lceil \kappa_{\rm TR,S}^{\rm 2nd} \max[\epsilon_g^{-2}\epsilon_H^{-1}, \epsilon_H^{-3}]\rceil$.*

Proof. Consider an iteration k of the trust-region algorithm (before it terminates). Then either $\|g_k\| > \epsilon_g$ or $\tau_k < -\epsilon_H$. In the first of these cases, (4.7), (4.11) and (4.13) yield that

$$-m_k(s_k) \geq \frac{1}{2}\epsilon_g \min\left[\frac{\epsilon_g}{\kappa_{\rm B}}, \kappa_\Delta \min[\epsilon_g, \epsilon_H]\right] = \frac{1}{2}\kappa_\Delta \epsilon_g \min[\epsilon_g, \epsilon_H],$$

while we obtain, in the second case, that

$$-m_k(s_k) \geq \frac{1}{2}\kappa_{\rm snc}|\tau_k|\Delta_k^2 \geq \frac{1}{2}\kappa_{\rm snc}\kappa_\Delta^2 \epsilon_H \min[\epsilon_g, \epsilon_H]^2$$

from (4.8) and (4.11). We thus obtain, using A.2 and the monotonically decreasing nature of the sequence $\{f(x_k)\}$, that

$$\begin{aligned} f(x_0) - f_{\rm low} &\geq \sum_{k=0}^{\infty} [f(x_k) - f(x_{k+1})] \\ &\geq \sum_{k=0, k\in\mathcal{S}} [f(x_k) - f(x_{k+1})] \\ &\geq \frac{1}{2}\eta_1 \sum_{k=0, k\in\mathcal{S}} \min[\kappa_\Delta \epsilon_g \min[\epsilon_g, \epsilon_H], \kappa_{\rm snc}\kappa_\Delta^2 \epsilon_H \min[\epsilon_g, \epsilon_H]^2] \\ &\geq \frac{1}{2}\eta_1 \kappa_{\rm snc}\kappa_\Delta^2 \sum_{k=0, k\in\mathcal{S}} \min[\epsilon_g, \epsilon_H] \min[\epsilon_g, \epsilon_H\epsilon_g, \epsilon_H^2] \\ &= \frac{1}{2} N_{\rm TR,S}^{\rm 2nd} \eta_1 \kappa_{\rm snc}\kappa_\Delta^2 \epsilon_H \min[\epsilon_g, \epsilon_H]^2 \end{aligned} \tag{4.14}$$

where $N_{\rm TR,S}^{\rm 2nd}$ is the total number of successful or very successful iterations such that (1.3) fails, and where we used the inequalities $\kappa_\Delta < 1$, $\kappa_{\rm snc} \leq 1$ and $\max[\epsilon_g, \epsilon_H] < 1$. The desired conclusion follows from this last inequality with

$$\kappa_{\rm TR,S}^{\rm 2nd} \stackrel{\text{def}}{=} \frac{2(f(x_0) - f_{\rm low})}{\eta_1 \kappa_{\rm snc}\kappa_\Delta^2}. \quad \square$$

Before concluding, we still need an analogue of Lemma 2.6 for the trust-region algorithm. Such a result is also described in [10], but we formalize it for the sake of clarity.

Lemma 4.6. *Suppose that* A.1 *and* A.3 *hold and, for any fixed $j \geq 0$, let $\mathcal{S}_j$ and $\mathcal{U}_j$ be defined in (2.12). Then one has that*

$$|\mathcal{U}_j| \leq \left\lceil \frac{\log \gamma_3}{|\log \gamma_2|} |\mathcal{S}_j| + \frac{1}{|\log \gamma_2|} \log \left(\frac{\Delta_0}{\kappa_\Delta \min[\epsilon_g, \epsilon_H]} \right) \right\rceil . \tag{4.15}$$

Proof. It follows from the mechanism of the trust-region algorithm that

$$\Delta_{k+1} \leq \gamma_3 \Delta_k \quad \text{for all } k \in \mathcal{S}_j \quad \text{and} \quad \Delta_{k+1} \leq \gamma_2 \Delta_k \quad \text{for all } k \in \mathcal{U}_j.$$

Thus we obtain that

$$\Delta_j \leq \Delta_0 \gamma_2^{|\mathcal{U}_j|} \gamma_3^{|\mathcal{S}_j|}.$$

But Lemma 4.4 gives that, as long as the trust-region algorithm has not terminated, (4.11) must hold. Therefore, we obtain that

$$|\mathcal{S}_j| \log \gamma_3 + |\mathcal{U}_j| \log \gamma_2 \geq \log \left(\frac{\kappa_\Delta \min[\epsilon_g, \epsilon_H]}{\Delta_0} \right).$$

Reorganizing this inequality using $\gamma_2 < 1$ and taking into account that $|\mathcal{U}_j|$ is an integer then yields (4.15). □

We may now state the final worst-case complexity bound for convergence of the trust-region algorithm to approximate second-order critical points.

Theorem 4.7. *Suppose that* A.1–A.3 *hold. Then, the trust-region algorithm produces an iterate x_k satisfying (1.3) (and thus terminates) after at most*

$$\lceil \kappa_{\rm TR,S}^{\rm 2nd} \max[\epsilon_g^{-2} \epsilon_H^{-1}, \epsilon_H^{-3}] \rceil \tag{4.16}$$

successful iterations and at most

$$\lceil \kappa_{\rm TR}^{\rm 2nd} \max[\epsilon_g^{-2} \epsilon_H^{-1}, \epsilon_H^{-3}] \rceil \tag{4.17}$$

iterations in total, where $\kappa_{\rm TR,S}^{\rm 2nd}$ and $\kappa_{\rm TR}^{\rm 2nd}$ are positive constants.

Proof. The first part of the theorem immediately results from Lemma 4.5. The second bound follows by applying Lemma 4.6 and noting that the term in $\log(1/\epsilon)$ arising from the second term on the left-hand side of (4.15) is dominated by the first as, obviously, $\log(1/\epsilon) = O(\epsilon^{-3})$ for $\epsilon \in (0, 1)$. □

As for the ARC algorithm, we now show that the bound stated in Theorem 4.7 cannot be improved. Again this is achieved by exhibiting a unidimensional example where this bound is attained. The example is itself a modification of that introduced in Section 3 and uses the definitions of x_0, g_k and $B_k = H_k = \tau_k$ given by (3.1). We now define

$$s_k = \Delta_k = \left(\frac{1}{k+1} \right)^{\frac{1}{3}+\delta} \tag{4.18}$$

(which gives the same steps as in Section 3) and

$$f(x_0) = \frac{1}{4}(\eta_1 + \eta_2)\zeta(1+3\delta) \quad \text{and} \quad f_{k+1} = f_k - \frac{1}{4}(\eta_1 + \eta_2) \left(\frac{1}{k+1} \right)^{1+3\delta} \tag{4.19}$$

and therefore the sequence $\{f_k\}$ is bounded below by zero. It is also clear from the derivation of the example in Section 3 that we may use Hermite interpolation to define the objective function f on $\mathbb{R}$ such that it is twice continuously differentiable and has a Lipschitz continuous Hessian. It therefore satisfies both A.1 and A.2. In order to verify that these functions and step values may be generated by a trust-region algorithm, we first note, using (3.1), (4.18) and (4.19), that

$$f_{k+1} = f_k - \frac{1}{4}(\eta_1 + \eta_2)|\tau_k|\Delta_k^2.$$

Hence we obtain[4] from (2.10) that

$$\rho_k = \frac{\frac{1}{4}(\eta_1 + \eta_2)|\tau_k|\Delta_k^2}{\frac{1}{2}|\tau_k|\Delta_k^2} = \frac{1}{2}(\eta_1 + \eta_2),$$

for each $k \geq 0$. Every iteration is therefore successful (but not very successful). According to (4.4), we may then choose Δ_{k+1} in the range $[\gamma_2\Delta_, \Delta_k]$ and our choice

$$\Delta_{k+1} = \left(\frac{k+1}{k+2}\right)^{\frac{1}{3}+\delta} \Delta_k$$

is thus acceptable assuming, without loss of generality, that $\gamma_2 \leq (\frac{1}{2})^{\frac{1}{3}+\delta}$.

As in Section 3, we have constructed an objective function f satisfying A.1–A.2 on which the trust-region algorithm will need, for any ϵ_g, ϵ_H and δ in $(0, 1)$, at least of the order of $O(\epsilon_H^{-3/(1+3\delta)})$ successful iterations to achieve approximate second-order optimality. The bounds given by (4.16) and (4.17) are therefore sharp when $\epsilon_g \geq \epsilon_H$. We have not been able to show that these bounds are sharp whenever $\epsilon_g \leq \epsilon_H$.

We conclude this paper by comparing the bounds for achieving (1.3) given for the ARC algorithm by (2.29) in Theorem 2.9 and for the trust-region algorithm by (4.16)–(4.17) in Theorem 4.7.

- If one assumes that $\epsilon_H \leq \epsilon_g$, then the two sets of bounds are qualitatively identical,[5] and we have seen that both are sharp.
- If $\epsilon_g < \epsilon_H < \epsilon_g^{1/2}$, then the worst-case bound for the trust-region method is $O(\epsilon_g^{-2}\epsilon_H^{-1}) = O(\epsilon_H^{-\theta})$ iterations at most for some $\theta \in (3, 5)$, while the corresponding (sharp) bound for the ARC algorithm remains $O(\epsilon_H^{-3})$, which is more favourable.
- Finally, if $\epsilon_g^{1/2} \leq \epsilon^H$, the worst-case bound for the trust-region method is now $O(\epsilon_g^{-2}\epsilon_H^{-1}) = O(\epsilon_g^{-5/2})$ iterations at most, but Cartis et al. [4] show that it is also at least $O(\epsilon_g^{-2})$. By comparison, the worst-case bound for the ARC algorithm is shown to be no worse than $O(\epsilon_g^{-2})$, while if (2.9) holds this improves to $O(\epsilon_g^{-3/2})$, which, according to Cartis et al. [4] is sharp. The choice of ϵ_H of the order of the square root of ϵ_g (which falls at the limit between this third case and the second) makes sense if one wishes to ensure independence of the stopping rule (1.3) from the effect of linear transformations of the problem's variables, and we note that such a choice is also implied by the definition of the measure of local optimality in [15].

We therefore see that the ARC algorithm has equal or better worst-case bounds than the trust-region algorithm in all cases, and that the difference is largest for the most practically relevant choice of the relative sizes of the first- and second-order stopping tolerances.

We conclude this section by observing that both presented examples are independent of the value of ϵ_g relative to ϵ_H, disentangling the interaction between the first- and second-order optimality measures. In particular, this is notable for the trust-region case, where Lemma 4.4 implies a strong interaction between the measures, reflected in Theorem 4.7. Note however, that in both Lemma 4.4 and in Theorem 4.7, if $\|g_k\| \leq \epsilon_g$ for all k (which is the case of our example), then it must be that $\tau_k < -\epsilon_H$ for all k until termination. Furthermore, then (4.11) becomes $\Delta_k \geq \kappa_\Delta\epsilon_H$, only the second-order model decrease applies in the proof of Lemma 4.5 and depends entirely on ϵ_H, yielding an upper bound of order ϵ_H^{-3} for the evaluation complexity of trust region. Thus, for the particular case when

4 Note that $\kappa_{\text{snc}} = 1$ because our example is unidimensional.

5 The constants differ.

only the curvature condition needs to be satisfied, this upper bound is sharp for the trust-region algorithm. (Similarly, when only the size of the gradient needs to be decreased, Theorem 4.7 yields an upper bound of order ϵ_g^{-2}, which was shown in [4] to be sharp for trust region.) These remarks illustrate that it is not just the relationship between ϵ_g and ϵ_H which matters for the worst-case bounds, but also how "close" $\|g_k\|$ and $|\tau_k|$ are to these thresholds.

5. Summary and perspectives

We have considered the worst-case complexity of achieving approximate second-order optimality for the ARC and trust-region algorithms. We have started by showing that the known bound of $O(\epsilon_H^{-3})$ ARC iterations can be derived for a variant of the algorithm not requiring multi-dimensional global optimization, and have then shown that the obtained bound is sharp. In addition, we have proved that a bound of the same type also holds for the standard trust-region algorithm, and that this second bound is also sharp whenever $\epsilon_H = O(\epsilon_g)$. We finally showed that the worst-case bound for the ARC algorithm is always as good or better than that for the trust-region method.

An obvious next step is to extend the worst-case analysis for second-order optimality to finite-difference and derivative-free schemes, in the spirit of Cartis et al. [8], and to constrained problems, possibly working along the lines of Cartis et al. [6]. It is also interesting to verify if the optimality properties of the ARC algorithm for convergence to approximate first-order point [5] can be extended to the ARC algorithm for the second-order case.

Acknowledgments

The work of the second author is funded by EPSRC Grant EP/E053351/1. All three authors are grateful to the Royal Society for its support through the International Joint Project 14265. Two anonymous referees also helped to improve the paper's presentation.

References

[1] A. Agarwal, P.L. Bartlett, P. Ravikummar, M.J. Wainwright, Information-theoretic lower bounds on the oracle complexity of convex optimization, in: Proceedings of the 23rd Annual Conference on Neural Information Processing Systems, 2009.
[2] C. Cartis, N.I.M. Gould, Ph.L. Toint, Adaptive cubic regularisation methods for unconstrained optimization. Part I: motivation, convergence and numerical results, Mathematical Programming. Series A 127 (2) (2011) 245–295.
[3] C. Cartis, N.I.M. Gould, Ph.L. Toint, Adaptive cubic regularisation methods for unconstrained optimization. Part II: worst-case function-evaluation complexity, Mathematical Programming. Series A (2010) doi:10.1007/s10107-009-0337-y.
[4] C. Cartis, N.I.M. Gould, Ph.L. Toint, On the complexity of steepest descent, Newton's and regularized Newton's methods for nonconvex unconstrained optimization, SIAM Journal on Optimization 20 (6) (2010) 2833–2852.
[5] C. Cartis, N.I.M. Gould, Ph.L. Toint, Optimal Newton-type methods for nonconvex smooth optimization problems, ERGO Technical Report 11-009, School of Mathematics, University of Edinburgh, United Kingdom, 2011.
[6] C. Cartis, N.I.M. Gould, Ph.L. Toint, An adaptive cubic regularisation algorithm for nonconvex optimization with convex constraints and its function-evaluation complexity, Technical Report 08/05, Department of Mathematics, FUNDP—University of Namur, Namur, Belgium, 2008.
[7] C. Cartis, N.I.M. Gould, Ph.L. Toint, Trust-region and other regularisation of linear least-squares problems, BIT 49 (1) (2009) 21–53.
[8] C. Cartis, N.I.M. Gould, Ph.L. Toint, On the oracle complexity of first-order and derivative-free algorithms for smooth nonconvex minimization, Technical Report naXys-03-2010, Namur Centre for Complex Systems, naXys, FUNDP-University of Namur, Namur, Belgium, 2010.
[9] A.R. Conn, N.I.M. Gould, Ph.L. Toint, Trust-Region Methods, in: MPS-SIAM Series on Optimization, vol. 01, SIAM, Philadelphia, USA, 2000.
[10] S. Gratton, A. Sartenaer, Ph.L. Toint, Recursive trust-region methods for multiscale nonlinear optimization, SIAM Journal on Optimization 19 (1) (2008) 414–444.
[11] 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, Cambridge, United Kingdom, 1981.
[12] A.S. Nemirovski, Efficient methods in convex programming, 1994. Lectures Notes (online) available on: http://www2.isye.gatech.edu/~nemirovs/OPTI_LectureNotes.pdf.
[13] Yu. Nesterov, Introductory Lectures on Convex Optimization, in: Applied Optimization, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2004.
[14] Yu. Nesterov, Accelerating the cubic regularization of Newton's method on convex problems, Mathematical Programming. Series A 112 (1) (2008) 159–181.

[15] Yu. Nesterov, B.T. Polyak, Cubic regularization of Newton method and its global performance, Mathematical Programming. Series A 108 (1) (2006) 177–205.
[16] S.A. Vavasis, Nonlinear Optimization: Complexity Issues, in: International Series of Monographs on Computer Science, Oxford University Press, Oxford, England, 1992.
[17] S.A. Vavasis, Approximation algorithms for indefinite quadratic programming, Mathematical Programming 57 (2) (1992) 279–311.
[18] S.A. Vavasis, Black-box complexity of local minimization, SIAM Journal on Optimization 3 (1) (1993) 60–80.
[19] L.N. Vicente, Worst case complexity of direct search, Technical Report, Department of Mathematics University of Coimbra Coimbra, Portugal, 2010.
[20] M. Weiser, P. Deuflhard, B. Erdmann, Affine conjugate adaptive Newton methods for nonlinear elastomechanics, Optimization Methods and Software 22 (3) (2007) 413–431.