Adaptive Regularization Algorithms with Inexact Evaluations for Nonconvex Optimization

Adaptive Regularization Algorithms with Inexact Evaluations for Nonconvex Optimization

Stefania Bellavia*, Gianmarco Gurioli†, Benedetta Morini‡, and Philippe L. Toint§

18 April 2019

Abstract

A regularization algorithm using inexact function values and inexact derivatives is proposed and its evaluation complexity analyzed. This algorithm is applicable to unconstrained problems and to problems with inexpensive constraints (that is constraints whose evaluation and enforcement has negligible cost) under the assumption that the derivative of highest degree is β-Hölder continuous. It features a very flexible adaptive mechanism for determining the inexactness which is allowed, at each iteration, when computing objective function values and derivatives. The complexity analysis covers arbitrary optimality order and arbitrary degree of available approximate derivatives. It extends results of Cartis, Gould and Toint [*Sharp worst-case evaluation complexity bounds for arbitrary-order nonconvex optimization with inexpensive constraints*, arXiv:1811.01220, 2018] on the evaluation complexity to the inexact case: if a q-th order minimizer is sought using approximations to the first p derivatives, it is proved that a suitable approximate minimizer within ϵ is computed by the proposed algorithm in at most $O\big(\epsilon^{-\frac{p+\beta}{p-q+\beta}}\big)$ iterations and at most $O\big(|\log(\epsilon)|\epsilon^{-\frac{p+\beta}{p-q+\beta}}\big)$ approximate evaluations. An algorithmic variant, although more rigid in practice, can be proved to find such an approximate minimizer in $O\big(|\log(\epsilon)| + \epsilon^{-\frac{p+\beta}{p-q+\beta}}\big)$ evaluations. While the proposed framework remains so far conceptual for high degrees and orders, it is shown to yield simple and computationally realistic inexact methods when specialized to the unconstrained and bound-constrained first- and second-order cases. The deterministic complexity results are finally extended to the stochastic context, yielding adaptive sample-size rules for subsampling methods typical of machine learning.

Keywords: Evaluation complexity, regularization methods, inexact functions and derivatives, subsampling methods.

1 Introduction

Evaluation complexity of algorithms for nonlinear and possibly nonconvex optimization problems has been the subject of active research in recent years. This field is concerned by de-

*Dipartimento di Ingegneria Industriale, Università degli Studi, Firenze, Italy. Member of the INdAM Research Group GNCS. Email: stefania.bellavia@unifi.it.

†Dipartimento di Matematica e Informatica "Ulisse Dini", Università degli Studi, Firenze, Italy. Member of the INdAM Research Group GNCS. Email: gianmarco.gurioli@unifi.it.

‡Dipartimento di Ingegneria Industriale, Università degli Studi, Firenze, Italy. Member of the INdAM Research Group GNCS. Email: benedetta.morini@unifi.it.

§Namur Center for Complex Systems (naXys), University of Namur, 61, rue de Bruxelles, B-5000 Namur, Belgium. Email: philippe.toint@unamur.be.

riving formal bounds on the number of evaluations of the objective function (and possibly of its derivatives) necessary to obtain approximate optimal solutions within a user-specified accuracy. Until recently, the results had focused on methods using first- and second-order derivatives of the objective function, and on convergence guarantees to first- or second-order stationary points [29, 23, 24, 19, 11]. Among these contributions, [24, 11] analyzed the "regularization method", in which a model of the objective function around a given iterate is constructed by adding a regularization term to the local Taylor expansion, model which is then approximately minimized in an attempt to find a new point with a significantly lower objective function value [21]. Such methods have been shown to possess optimal evaluation complexity [14] for first- and second-order models and minimizers, and have generated considerable interest in the research community. A theoretically significant step was made in [7] for unconstrained problems, where evaluation complexity bounds were obtained for convergence to first-order stationary points of a simplified regularization method using models of arbitrary degree. Even more recently, [13] proposed a conceptual unified framework subsuming all known results for regularization methods, establishing an upper evaluation complexity bound for arbitrary model degree and also, for the first time, for arbitrary orders of optimality. This paper additionally covers unconstrained problems and problems involving "inexpensive" constraints, that is constraints whose evaluation/enforcement cost is negligible compared to that of evaluating the objective function and its derivatives. It also allows for a full range of smoothness assumptions on the objective function. Finally it proves that the complexity results obtained are optimal in the sense that upper and lower evaluation complexity bounds match in order. In [13], all the above mentioned results are established for versions of the regularization algorithms where it is assumed that objective function values and values of its derivatives (when necessary) can be computed exactly.

In practice, it may sometimes be difficult or impossible to obtain accurate values of the problem's function and/or derivatives. This difficulty has been known for a long time and has generated its own stream of results, among which we note the trust-region method using dynamic accuracy on the objective function and (possibly on) its gradient (see Sections 8.4.1.1 and 10.6 of [17] and [4]), and the purely probabilistic approaches of [25] and [8]. Since unconstrained cubic regularization methods have become popular in the machine learning community (see [1] for a survey of optimization in this area) due to their optimal complexity, several contributions have considered building those function and derivative's approximations by "subsampling" the (very many) nonlinear terms whose sum defines the objective functions typical of machine learning applications. Inexact Hessian information is considered in [16, 5, 30, 31], approximate gradient and Hessian evaluations are used in [12, 15, 27, 32], function, gradient and Hessian values are sampled in [22, 6]. The amount of inexactness allowed is controlled dynamically in [12, 15, 22, 16, 5].

Contributions. The present paper proposes an extension of the unifying framework of [13] for unconstrained or inexpensively-constrained problems that allows inexact evaluations of the objective function and of the required derivatives, in an adaptive way inspired by the trust-region scheme of [17, Section 10.6]. This extension has the advantage of preserving the optimal complexity of the standard regularization methods and, as in [13], evaluation complexity results are provided for arbitrary model degree and arbitrary order of optimality. In particular, the proposed framework allows all combinations of exact/inexact objective functions and derivatives of any order (including of course degrees and orders one and two, for which simple specializations are outlined). We also consider an interesting but practically

more restrictive variant of our algorithm for which an improved complexity can be derived. We finally consider a stochastic version of our framework and derive rules for sample size in the context of subsampling methods for machine learning.

The paper is organized as follows. Section 2 recalls the notions of high-order optimality proposed in [13] and introduces the general Adaptive Regularization algorithm with model of order p allowing Dynamic Accuracy (ARpDA). The details of how to obtain the desired relative accuracy levels from known absolute errors are examined in Section 3. The evaluation complexity of obtaining approximate minimizers using this algorithm is then analyzed in Section 4. The algorithmic variant of the algorithm is discussed in Section 5. The general framework is specialized to first- and second-order optimization in Section 6, showing that practical implementation for low order is simple and computationally realistic. The stochastic evaluation complexity and sampling rules for machine learning applications are finally derived in Section 7. Conclusions and perspectives are presented in Section 8.

Notations. Unless otherwise specified, $\|\cdot\|$ denotes the standard Euclidean norm for vectors and matrices. For a general symmetric tensor S of order p, we define

$$\|S\|_{[p]} \stackrel{\rm def}{=} \max_{\|v\|=1} |S[v]^p| = \max_{\|v_1\|=\cdots=\|v_p\|=1} |S[v_1,\ldots,v_p]| \tag{1.1}$$

the induced Euclidean norm. We also denote by $\nabla_x^j f(x)$ the j-th order derivative tensor of f evaluated at x and note that such a tensor is always symmetric for any $j \geq 2$. $\nabla_x^0 f(x)$ is a synonym for $f(x)$. $\lceil\alpha\rceil$ and $\lfloor\alpha\rfloor$ denote the smallest integer not smaller than α and the largest integer not exceeding α, respectively. If i is a non-negative integer and β a real in $(0,1]$ we define $(i+\beta)! = \prod_{\ell=1}^{i}(\ell+\beta)$. For symmetric matrices, $\lambda_{\min}[M]$ is the leftmost eigenvalue of M. Pr[event] finally denotes the probability of an event. Finally $\mathrm{globmin}_{x\in\mathcal{S}}\, f(x)$ denotes the smallest value of $f(x)$ over $x \in \mathcal{S}$.

2 High-order necessary conditions and the ARpDA algorithm

Given $p \geq 1$, we consider the set-constrained optimization problem

$$\min_{x\in\mathcal{F}} f(x), \tag{2.1}$$

where $\mathcal{F} \subseteq \mathbb{R}^n$ is closed and nonempty, and where we assume that the *values of the objective function f and its derivatives must be computed inexactly.* We also assume that $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n)$, meaning that:

- f is p-times continuously differentiable,
- f is bounded below by $f_{\rm low}$, and
- the p-th derivative tensor of f at x is globally Hölder continuous, that is, there exist constants $L \geq 0$ and $\beta \in (0,1]$ such that, for all $x, y \in \mathbb{R}^n$,

$$\|\nabla_x^p f(x) - \nabla_x^p f(y)\|_{[p]} \leq L\|x-y\|^\beta. \tag{2.2}$$

The more standard case where f is assumed to have Lipschitz-continuous p-th derivative is recovered by setting $\beta = 1$ in the above assumptions (for example, the choices $p = 2$ and

$\beta = 1$ correspond to the assumption that f has a Lipschitz continuous Hessian). In what follows, we assume that β is known.

If we denote the pth degree Taylor expansion of f around x evaluated at s by

$$T_p^f(x,s) \stackrel{\rm def}{=} f(x) + \sum_{\ell=1}^{p} \frac{1}{\ell!} \nabla_x^\ell f(x)[s]^\ell, \tag{2.3}$$

we may then define the *Taylor increment* by

$$\Delta T_p^f(x,s) = T_p^f(x,0) - T_p^f(x,s). \tag{2.4}$$

Under the above assumptions, we recall the crucial bounds on differences between f and its derivatives and their Taylor's expansion.

Lemma 2.1 [13, Lemma 2.1] Let $f \in C^{p,\beta}(\mathbb{R}^n)$, and $T_p^f(x,s)$ be the Taylor approximation of $f(x+s)$ about x given by (2.3). Then for all $x, s \in \mathbb{R}^n$,

$$|f(x+s) - T_p^f(x,s)| \leq \frac{L}{(p+\beta)!} \|s\|^{p+\beta}, \tag{2.5}$$

$$\|\nabla_x^j f(x+s) - \nabla_s^j T_p^f(x,s)\|_{[j]} \leq \frac{L}{(p-j+\beta)!} \|s\|^{p+\beta-j}. \quad (j = 1, \ldots, p). \tag{2.6}$$

We also follow [13] and define a q-th-order-necessary minimizer as a point $x \in \mathbb{R}^n$ such that, for some $\delta \in (0,1]$,

$$\phi_{f,q}^\delta(x) \stackrel{\rm def}{=} f(x) - \operatorname*{globmin}_{\substack{x+d\in\mathcal{F} \\ \|d\|\leq\delta}} T_q^f(x,d) = 0. \tag{2.7}$$

Observe that, in the unconstrained case, this definition subsumes the usual optimality criteria for orders one and two, since, if $q = 1$, (2.7) gives that, for any $\delta \in (0,1]$ (and in particular for $\delta = 1$),

$$\phi_{f,q}^\delta(x) = \|\nabla_x^1 f(x)\|\delta, \tag{2.8}$$

and first-order optimality is thus equivalent to

$$\|\nabla_x^1 f(x)\| = 0.$$

Similarly, for $q = 2$, (2.7) is equivalent to

$$\|\nabla_x^1 f(x)\| = 0 \quad \text{and} \quad \lambda_{\min}[\nabla_x^2 f(x)] \geq 0. \tag{2.9}$$

Its properties are further discussed in [13], but we emphasize that, for any $q \geq 1$ and in contrast with other known measures, it varies continuously when x varies continuosly in $\mathcal{F}$. In the unconstrained case, solving the global optimization problem involved in its definition is easy for $q = 1$ as the global minimizer is analytically given by $d_* = -\delta\, \nabla_x^1 f(x)/\|\nabla_x^1 f(x)\|$, and also for $q = 2$ using a trust-region scheme (whose cost is essentially comparable to that

of computing the leftmost eigenvalue in (2.9)). However this task may become NP-hard for larger q. This makes $\phi^\delta_{f,q}(x)$ an essentially theoretical tool for these cases. In any case, the computation of $\phi^\delta_{f,q}(x)$ does not involve evaluating f or any of its derivatives, and its cost therefore does not affect the evaluation complexity of interest here.

If we now relax the notion of exact minimizers, we may define an (ϵ,δ)-approximate q-th-order-necessary minimizer as a point $x \in \mathbb{R}^n$

$$\phi^\delta_{f,q}(x) \leq \epsilon\chi_q(\delta), \tag{2.10}$$

where

$$\chi_q(\delta) \stackrel{\rm def}{=} \sum_{\ell=1}^{q} \frac{\delta^\ell}{\ell!} \tag{2.11}$$

provides a natural scaling. Again this notion reduces to familiar concepts in the low-order unconstrained cases. For instance, we verify that for unconstrained problems with $q = 2$, (2.10) requires that, if d is the global minimizer in (2.7) (the solution of a trust-region problem),

$$\max\left[0, -\left(\nabla^1_x f(x)^T d + \tfrac{1}{2} d^T \nabla^2_x f(x) d\right)\right] \leq \epsilon(\delta + \tfrac{1}{2}\delta^2),$$

which automatically holds for any $\delta \in (0,1]$ if $\|\nabla^1_x f(x)\| \leq \epsilon$ and $\lambda_{\min}[\nabla^2_x f(x)] \geq -\epsilon$. We note that, when assessing whether x is an (ϵ,δ)-approximate q-th-order-necessary minimizer, the global minimization in (2.7) can be stopped as soon as $\Delta T^f_q(x,d)$ exceeds $\epsilon\chi_q(\delta)$, thereby significantly reducing the cost of this assessment.

Having defined what we mean by high-order approximate minimizers, we now turn to describing what we mean by inaccurate objective function and derivatives values. It is important to observe at this point that, in an optimization problem, the role of the objective function is more central than that of any of its derivatives, since it is the quantity we ultimately wish to decrease. For this reason, we will handle the allowed inexactness in f differently from that in $\nabla^j_x f$: we will require an (adaptive) *absolute* accuracy for the first and a *relative* accuracy for the second. In fact, we can, in a first approach, abstract the relative accuracy requirements for the derivatives $\nabla^j_x f(x)$ into a requirement on the relative accuracy of $\Delta T^f_p(x,s)$. Let $\omega \in [0,1]$ represent a relative accuracy level and denote inexact quantities with an overbar. For what follows, we will thus require that, if

$$\overline{\Delta T}^f_p(x,s,\omega) = \overline{T}^f_p(x_k,0,\omega) - \overline{T}^f_p(x_k,s,\omega), \tag{2.12}$$

then

$$|\overline{\Delta T}^f_p(x,s,\omega) - \Delta T^f_p(x,s)| \leq \omega \overline{\Delta T}^f_p(x,s,\omega). \tag{2.13}$$

It may not be obvious at this point how to enforce this relative error bound: this is the object of Section 3 below. For now, we simply assume that it can be done in a finite number of evaluations of $\{\overline{\nabla^j_x f(x)}\}^p_{j=1}$ which are inexact approximations of $\{\nabla^j_x f(x)\}^p_{j=1}$.

Given an inexactly computed $\overline{\Delta T}^f_p(x,s,\omega)$ satisfying (2.13), we then have to consider to compute our optimality measure inexactly too. Observing that the definition (2.7) is independent of $f(x)$ because of cancellation, we see that

$$\overline{\phi}^\delta_{f,q}(x,\omega) = \max\left[0, \operatorname*{globmax}_{\substack{x+d\in\mathcal{F}\\ \|d\|\leq\delta}} \overline{\Delta T}^f_q(x,d,\omega)\right]. \tag{2.14}$$

Under the above assumptions, we now describe an algorithm allowing inexact computation of both the objective function and its derivatives whose purpose is to find (for given q and a suitable relative accuracy ω) a point x_k satisfying

$$\overline{\phi}^{\delta}_{f,q}(x,\omega) \leq \frac{\epsilon}{1+\omega}\chi_q(\delta) \tag{2.15}$$

for some optimality radius $\delta \in (0,1]$. This algorithm uses a regularized Taylor's model defined at iteration k by

$$m_k(s) \stackrel{\rm def}{=} \overline{T}^f_p(x_k,s,\omega_k) + \frac{\sigma_k}{(p+\beta)!}\|s\|^{p+\beta}. \tag{2.16}$$

This model is then approximately minimized and the resulting trial point is then accepted or rejected depending on whether or not it produces a significant decrease. This is detailed in Algorithm 2.1 on the following page.

Some comments on this algorithm are useful at this stage.

1. That Step 2 may not be able, for $q > 2$, to compute a nonzero step (and should then cause termination) can be seen by considering the following one-dimensional example. Let $p = q = 3$, $\mathcal{F} = \mathbb{R}$, $\omega_k = 0$ and $\delta_{k-1} = 1$ and suppose that $T_3(x_k,s) = s^2 - 2s^3$ and also that $\sigma_k = 24$. This implies that $m_k(s) = s^2 - 2s^3 + s^4 = s^2(1-s)^2$ and we immediately see that the origin is a global minimizer of $m_k(s)$. But a simple calculation shows that $\phi^{\delta_{k-1}}_{f,q} = T_3(x_k,0) - T_3(x_k,1) = 1$ and hence termination will not occur in Step 1 if $\epsilon < 1/\chi_3(1) = 4/7$. As a consequence, as was pointed out in [13], the possibility of a zero s_k cannot be ignored in Step 2. In this case, it is not possible to satisfy (2.19) and the algorithm terminates with $x_\varepsilon = x_k$. It has been proved in [13, Lemma 2.6] that this is acceptable (see also Lemma 2.4 below).

2. Our assumption (2.13) is used three times in the algorithm: in Step 1 for computing $\overline{\phi}^{\delta_{k-1}}_{f,q}(x_k,\omega_k)$ and in Step 2 when computing s_k and $\overline{\phi}^{\delta_k}_{m_k,q}(s_k,\omega_k)$.

3. As indicated above, we require a bound on the absolute error in the objective function value: this is the object of (2.21) and (2.22), where we introduced the notation $\overline{f}_k(x_k,\omega_k)$ to denote an inexact approximation of $f(x_k)$. Note that a new value of $\overline{f}_k(x_k,\omega_k)$ should be computed to ensure (2.22) in Step 3 only if $k > 0$ and $\omega_{k-1}\overline{\Delta T}^f_p(x_{k-1},s_{k-1},\omega_{k-1}) > \omega_k\overline{\Delta T}^f_p(x_k,s_k,\omega_k)$. If this is the case the (inexact) function value is computed twice per iteration instead of just once.

4. At variance with the trust-region method with dynamic accuracy of [17, Section 10.6] and [4], we do not recompute approximate values of the objective function at x_k once the computation of s_k is complete (provided we can ensure (2.13), as discussed in Section 3).

5. If $\|s_k\| \geq \mu\epsilon^{\frac{1}{p-q+\beta}}$ in Step 2, then the (potentially costly) calculation of $\overline{\phi}^{\delta_k}_{m_k,q}(s_k,\omega_k)$ is unecessary and δ_k may be chosen arbitrarily in $(0,1]$.

6. We call iteration k *successful* when $\rho_k \geq \eta_1$ and $x_{k+1} = x_k + s_k$. The iteration is called *unsuccessful* otherwise, and $x_{k+1} = x_k$ in this case. We use the notation

$$\mathcal{S}_k = \{j \in \{0,\ldots,k\} \mid \rho_j \geq \eta_1\} \tag{2.26}$$

to denote the set of successful iterations of index at most k.

Algorithm 2.1: Adaptive Regularization of order p with Dynamic Accuracy (ARpDA)

Step 0: Initialization. An initial point $x_0 \in \mathcal{F}$ and an initial regularization parameter $\sigma_0 > 0$ are given, as well as an accuracy level $\epsilon \in (0,1)$ and an initial relative accuracy $\omega_0 \geq 0$. The constants κ_ω, δ_{-1}, θ, μ, η_1, η_2, γ_1, γ_2, γ_3 and $\sigma_{\min}$ are also given and satisfy $\theta > 0$, $\mu \in (0,1]$, $\delta_{-1} \in (0,1]$, $\sigma_{\min} \in (0, \sigma_0]$,

$$0 < \eta_1 \leq \eta_2 < 1, \;\; 0 < \gamma_1 < 1 < \gamma_2 < \gamma_3, \tag{2.17}$$

$$\alpha \in (0,1), \quad \kappa_\omega \in (0, \tfrac{1}{2}\alpha\eta_1] \;\; \text{and} \;\; \omega_0 = \min\left[\kappa_\omega, \frac{1}{\sigma_0}\right]. \tag{2.18}$$

Set $k = 0$.

Step 1: Compute the optimality measure and check for termination. Compute $\overline{\phi}_{f,q}^{\delta_{k-1}}(x_k, \omega_k)$. If (2.15) holds with $\delta = \delta_{k-1}$, terminate with the approximate solution $x_\epsilon = x_k$.

Step 2: Step calculation. Attempt to compute a step $s_k \neq 0$ such that $x_k + s_k \in \mathcal{F}$ and an optimality radius $\delta_k \in (0,1]$ by approximately minimizing the model $m_k(s)$ in the sense that

$$m_k(s_k) < m_k(0) \tag{2.19}$$

and

$$\|s_k\| \geq \mu\epsilon^{\frac{1}{p-q+\beta}} \quad \text{or} \quad \overline{\phi}_{m_k,q}^{\delta_k}(s_k, \omega_k) \leq \frac{\theta\|s_k\|^{p-q+\beta}}{(p-q+\beta)!}\,\chi_q(\delta_k). \tag{2.20}$$

If no such step exists, terminate with the approximate solution $x_\epsilon = x_k$.

Step 3: Acceptance of the trial point. Compute $\overline{f}_k(x_k + s_k, \omega_k)$ ensuring that

$$|\overline{f}_k(x_k + s_k, \omega_k) - f(x_k + s_k)| \leq \omega_k |\overline{\Delta T}_p^f(x_k, s_k, \omega_k)|. \tag{2.21}$$

Also ensure (by setting $\overline{f}_k(x_k, \omega_k) = \overline{f}_{k-1}(x_k, \omega_{k-1})$ or by (re)computing $\overline{f}_k(x_k, \omega_k)$) that

$$|\overline{f}_k(x_k, \omega_k) - f(x_k)| \leq \omega_k |\overline{\Delta T}_p^f(x_k, s_k, \omega_k)|. \tag{2.22}$$

Then define

$$\rho_k = \frac{\overline{f}_k(x_k, \omega_k) - \overline{f}_k(x_k + s_k, \omega_k)}{\overline{\Delta T}_p^f(x_k, s_k, \omega_k)}. \tag{2.23}$$

If $\rho_k \geq \eta_1$, then define $x_{k+1} = x_k + s_k$; otherwise define $x_{k+1} = x_k$.

Step 4: Regularization parameter update. Set

$$\sigma_{k+1} \in \begin{cases} [\max(\sigma_{\min}, \gamma_1\sigma_k), \sigma_k] & \text{if } \rho_k \geq \eta_2, \\ [\sigma_k, \gamma_2\sigma_k] & \text{if } \rho_k \in [\eta_1, \eta_2), \\ [\gamma_2\sigma_k, \gamma_3\sigma_k] & \text{if } \rho_k < \eta_1. \end{cases} \tag{2.24}$$

Step 5: Relative accuracy update. Set

$$\omega_{k+1} = \min\left[\kappa_\omega, \frac{1}{\sigma_{k+1}}\right]. \tag{2.25}$$

Increment k by one and go to Step 1.

7. As indicated above, ensuring (2.13) may require a certain number of (approximate) evaluations of the derivatives of f. For a single iteration of the algorithm, these evaluations are always at the current iterate x_k.

8. It is worth noting that from (2.17), (2.18), (2.24) and (2.25), together with the positivity of σ_0 and $\sigma_{\min}$,
$$0 < \omega_k \leq \kappa_\omega < 1. \tag{2.27}$$

We now state some properties of the algorithm that are derived without modification from the case where the computation of f and its derivatives are exact.

Lemma 2.2 [11, Theorem 2.1] The mechanism of the ARpDA algorithm ensures that, if

$$\sigma_k \leq \sigma_{\max}, \tag{2.28}$$

for some $\sigma_{\max} > 0$, then

$$k+1 \leq |\mathcal{S}_k| \left(1 + \frac{|\log \gamma_1|}{\log \gamma_2}\right) + \frac{1}{\log \gamma_2} \log\left(\frac{\sigma_{\max}}{\sigma_0}\right). \tag{2.29}$$

This shows that the number of unsuccessful iterations must remain a fixed proportion of that of the successful ones.

Lemma 2.3 [13, Lemma 2.5] Suppose that $s_k^* \neq 0$ is a global minimizer of $m_k(s)$ under the constraint that $x_k + s \in \mathcal{F}$, such $m_k(s_k^*) < m_k(0)$. Then there exist a neighbourhood of s_k^* and a range of sufficiently small δ such that (2.19) and the second part of (2.20) hold for any s_k in the intersection of this neighbourhood with $\mathcal{F}$ and any δ_k in this range.

This last lemma thus ensures that the algorithm is well-defined when $s_k \neq 0$. The lemma below shows that it is reasonable to terminate the algorithm whenever a nonzero descent step cannot be computed.

Lemma 2.4 [13, Lemma 2.6] Suppose that the algorithm terminates in Step 2 of iteration k with $x_\varepsilon = x_k$. Then there exists a $\delta \in (0, 1]$ such that (2.15) holds for $x = x_\varepsilon$.

3 Enforcing the relative error on Taylor increments

We now return to the question of enforcing (2.13). For improved readability, we temporarily ignore the iteration index k.

3.1 The accuracy checks

While there may be circumstances where (2.13) can be enforced directly, we consider here that the only control the user has on the accuracy of $\overline{\Delta T}_p^f(x, s, \omega)$ is by enforcing bounds $\{\varepsilon_j\}_{j=1}^p$ on the absolute errors on the derivative tensors $\{\nabla_x^j f(x)\}_{j=1}^p$. In other words, we seek to ensure (2.13) by selecting absolute accuracies $\{\varepsilon_j\}_{j=1}^p$ such that, when

$$\|\overline{\nabla_x^j f}(x) - \nabla_x^j f(x)\|_{[j]} \leq \varepsilon_j \;\text{ for }\; j \in \{1, \ldots, p\}, \tag{3.1}$$

the desired accuracy requirement follows.

In all cases described below, the process can be viewed as an iteration with four main steps. The first is to compute the relevant approximate derivative satisfying (3.1) for given values of $\{\varepsilon_j\}_{j=1}^p$. The second is to use these approximate derivatives to compute the desired Taylor increment and associated quantities. Tests are then performed in the third step to verify the desired accuracy requirements and terminate if they are met. If not the case, the absolute accuracies $\{\varepsilon_j\}_{j=1}^p$ are then decreased before a new iteration is started.

As can be expected, a suitable relative accuracy requirement will be achievable as long as $\overline{\Delta T}_p^f(x,s,\omega)$ remains safely away from zero, but, if exact computations are to be avoided, we may have to accept a simpler absolute accuracy guarantee when $\overline{\Delta T}_p^f(x,s,\omega)$ vanishes.

We then formalize the resulting accuracy tests in the VERIFY algorithm, stated as Algorithm 3.1 on the current page.

Assume that for a vector v_ω, a bound $\delta \geq \|v_\omega\|$, a degree r, the requested relative and absolute accuracies ω and $\xi > 0$, the increment $\overline{\Delta T}_r(x,v_\omega,\omega)$ are given. We intend to use the algorithm for $\overline{\Delta T}_q^f(x,v_\omega,\omega)$, $\overline{\Delta T}_p^f(x,v_\omega,\omega)$ and $\overline{\Delta T}_q^{m_k}(x,v_\omega,\omega)$. For keeping our development general, we use the notations $\overline{\Delta T}_r(x,v_\omega,\omega)$ and $\Delta T_r(x,v_\omega)$ without superscript. Moreover, we assume that the current absolute accuracies $\{\zeta_j\}_{j=1}^r$ of the derivatives of $\overline{T}_r(x,v_\omega,\omega)$ with respect to v_ω at $v_\omega = 0$ are given. Because it will be the case below, we assume for simplicity that $\overline{\Delta T}_r(x,v_\omega,\omega) \geq 0$.

Algorithm 3.1: Verify the accuracy of $\overline{\Delta T}_r(x,v_\omega,\omega)$

$$\texttt{flag} = \text{VERIFY}\Big(\delta, \overline{\Delta T}_r(x,v_\omega,\omega), \{\zeta_j\}_{j=1}^r, \omega, \xi\Big)$$

Set $\texttt{flag} = 0$.

- If
$$\overline{\Delta T}_r(x,v_\omega,\omega) = 0 \quad \text{and} \quad \max_{j\in\{1,\ldots,r\}} \zeta_j \leq \xi, \tag{3.2}$$
set $\texttt{flag} = 1$.
- Else, if
$$\overline{\Delta T}_r(x,v_\omega,\omega) > 0 \quad \text{and} \quad \sum_{j=1}^r \frac{\zeta_j}{j!}\delta^j \leq \omega\overline{\Delta T}_r(x,v_\omega,\omega), \tag{3.3}$$
set $\texttt{flag} = 2$.
- Else, if
$$\overline{\Delta T}_r(x,v_\omega,\omega) > 0 \quad \text{and} \quad \sum_{j=1}^r \frac{\zeta_j}{j!}\delta^j \leq \xi\chi_r(\delta), \tag{3.4}$$
set $\texttt{flag} = 3$.

Let us now consider what properties are ensured for the various possible values of `flag`.

Lemma 3.1 Suppose that

$$\left\| \left[\overline{\nabla^j_{v_\omega} T_r}(x, v_\omega)\right]_{v_\omega=0} - \left[\nabla^j_{v_\omega} T_r(x, v_\omega)\right]_{v_\omega=0} \right\|_{[j]} \leq \zeta_j \ \text{ for } \ j \in \{1, \ldots, r\} \tag{3.5}$$

and $\omega \in (0, 1)$. Then we have that

- if
$$\max_{j\in\{1,\ldots,r\}} \zeta_j \leq \xi, \tag{3.6}$$
then the VERIFY algorithm returns a nonzero `flag`,
- if the VERIFY algorithm terminates with $\texttt{flag} = 1$, then $\overline{\Delta T}_r(x, v_\omega, \omega) = 0$ and
$$\left|\overline{\Delta T}_r(x, v, \omega) - \Delta T_r(x, v)\right| \leq \xi\chi_r(\|v\|) \qquad \text{for all } v, \tag{3.7}$$
- if the VERIFY algorithm terminates with $\texttt{flag} = 2$, then $\overline{\Delta T}_r(x, v_\omega, \omega) > 0$ and
$$\left|\overline{\Delta T}_r(x, v, \omega) - \Delta T_r(x, v)\right| \leq \omega\overline{\Delta T}_r(x, v_\omega, \omega), \qquad \text{for all } v \text{ with } \|v\| \leq \delta, \tag{3.8}$$
- if the VERIFY algorithm terminates with $\texttt{flag} = 3$, then $\overline{\Delta T}_r(x, v_\omega, \omega) > 0$ and
$$\max\left[\overline{\Delta T}_r(x, v_\omega, \omega), \left|\overline{\Delta T}_r(x, v, \omega) - \Delta T_r(x, v)\right|\right] \leq \frac{\xi}{\omega}\chi_r(\delta) \ \text{ for all } v \text{ with } \|v\| \leq \delta. \tag{3.9}$$

Proof. W □

e first prove the first proposition. If $\overline{\Delta T}_r(x, v_\omega, \omega) = 0$ and (3.6), then (3.2) ensures that $\texttt{flag} = \texttt{1}$ is returned. If $\overline{\Delta T}_r(x, v_\omega, \omega) > 0$, from (2.11) and (3.6), we deduce that

$$\sum_{j=1}^{r} \frac{\zeta_j}{j!}\delta^j \leq \Big[\max_{j\in\{1,\ldots,r\}} \zeta_j\Big]\chi_r(\delta) \leq \xi\chi_r(\delta)$$

also causing termination with $\texttt{flag} = 3$ because of (3.4) if it has not occurred with $\texttt{flag} = 2$ because of (3.3), hence proving the first proposition.

Consider now the three possible termination cases and suppose first that termination occurs with $\texttt{flag} = 1$. Then, using the triangle inequality, (3.5), (3.2) and (2.11), we have that, for any v,

$$\left|\overline{\Delta T}_r(x, v, \omega) - \Delta T_r(x, v)\right| \leq \sum_{j=1}^{r} \frac{\zeta_j}{j!}\|v\|^j \leq \xi\chi_r(\|v\|)$$

yielding (3.7). Suppose now that `flag = 2`. Then (3.3) holds and for any v with $\|v\| \leq \delta$,

$$\left|\overline{\Delta T}_r(x, v, \omega) - \Delta T_r(x, v)\right| \leq \sum_{j=1}^{r} \frac{\zeta_j}{j!}\|v\|^j \leq \sum_{j=1}^{r} \frac{\zeta_j}{j!}\delta^j \leq \omega\overline{\Delta T}_r(x, v_\omega, \omega),$$

which is (3.8). Suppose finally that `flag = 3`. Since termination did not occur in (3.3), we have that

$$0 < \omega\overline{\Delta T}_r(x, v_\omega, \omega) \leq \xi\chi_r(\delta). \tag{3.10}$$

Furthermore, (3.4) implies that, for any v with $\|v\| \leq \delta$,

$$\left|\overline{\Delta T}_r(x,v,\omega) - \Delta T_r(x,v)\right| \leq \sum_{j=1}^{r} \frac{\zeta_j}{j!}\|v\|^j \leq \sum_{j=1}^{r} \frac{\zeta_j}{j!}\delta^j \leq \frac{\xi}{\omega}\chi_r(\delta).$$

This inequality and (3.10) together imply (3.9).
Clearly, the outcome corresponding to our initial aim to obtain a relative error at most ω corresponds to the case where `flag` $= 2$. As we will see below, the two other cases are also useful.

3.2 Computing $\overline{\phi}_{f,q}^{\delta_{k-1}}(x_k,\omega_k)$

We now consider, in Algorithm 3.2, how to compute the optimality measure $\overline{\phi}_{f,q}^{\delta_{k-1}}(x_k,\omega_k)$ in Step 1 of the ARpDA algorithm.

We immediately observe that Algorithm 3.2 terminates in a finite number of iterations, since it does so as soon as `flag` > 0, which, because of the first proposition of Lemma 3.1, must happen after a finite number of passes in iterations using (3.12). We discuss in Section 3.4 exactly how many such decreases might be needed.

We now verify that terminating the ARpDA algorithm as indicated in this modified version of Step 1 provides the required result. We start noting that, if x_k is an isolated feasible point (i.e. such that the intersection of any ball of radius $\delta_{k-1} > 0$ centered at x_k with $\mathcal{F}$ is reduced to x_k), then clearly $d_k = 0$ and thus, irrespective of ω_k and $\delta_{k-1} > 0$,

$$\phi_{f,q}^{\delta_{k-1}}(x_k) = 0 = \overline{\Delta T}_q^f(x_k,d_k,\omega_k) = \overline{\phi}_{f,q}^{\delta_{k-1}}(x_k,\omega_k), \tag{3.13}$$

which means that $\overline{\phi}_{f,q}^{\delta_{k-1}}(x_k,\omega_k)$ is a faithful indicator of optimality at x_k.

Lemma 3.2 If the ARpDA algorithm terminates within Step 1.4, then

$$\phi_{f,q}^{\delta_{k-1}}(x_k) \leq \epsilon\chi_q(\delta_{k-1}) \tag{3.14}$$

and x_k is a (ϵ,δ_{k-1})-approximate q-th-order-necessary minimizer. Otherwise Algorithm 3.2 terminates with

$$(1-\omega_k)\overline{\phi}_{f,q}^{\delta_{k-1}}(x_k,\omega_k) \leq \phi_{f,q}^{\delta_{k-1}}(x_k) \leq (1+\omega_k)\overline{\phi}_{f,q}^{\delta_{k-1}}(x_k,\omega_k). \tag{3.15}$$

Proof. W □

e first notice that Step 1.2 of Algorithm 3.2 yields (3.5) with $T_r = T_r^f$, $r = q$ and $\{\zeta_j\}_{j=1}^r = \{\varepsilon_{j,i_\varepsilon}\}_{j=1}^q$. Furthermore, $\omega = \omega_k \in (0,1)$, so that the assumptions of Lemma 3.1 are satisfied. If x_k is an isolated feasible point, the lemma's conclusions directly follow from (3.13). Assume therefore that x_k is not an isolated feasible point and note first that, because Step 1.3 finds the global maximum of $\overline{\Delta T}_q^f(x_k,d,\omega_k)$, we have that $\overline{\Delta T}_q^f(x_k,d_k,\omega_k) \geq 0$. Suppose now that,

Algorithm 3.2: Modified Step 1 of the ARpDA algorithm

Step 1: Compute the optimality measure and check for termination.

Step 1.0: The iterate x_k and the radius $\delta_{k-1} \in (0,1]$ are given, as well as constants $\gamma_\varepsilon \in (0,1)$ and $\kappa_\varepsilon > 0$. Set $i_\varepsilon = 0$.

Step 1.1: Choose an initial set of derivative absolute accuracies $\{\varepsilon_{j,0}\}_{j=1}^p$ such that

$$\varepsilon_{j,0} \leq \kappa_\varepsilon \quad \text{for} \quad j \in \{1,\ldots,p\}. \tag{3.11}$$

Step 1.2: If unavailable, compute $\{\overline{\nabla_x^j f}(x_k)\}_{j=1}^q$ satisfying

$$\|\overline{\nabla_x^j f}(x) - \nabla_x^j f(x)\|_{[j]} \leq \varepsilon_{j,i_\varepsilon} \ \text{ for } \ j \in \{1,\ldots,q\}.$$

Step 1.3: Solve

$$\operatorname*{globmax}_{\substack{x_k+d\in\mathcal{F}\\ \|d\|\leq\delta_{k-1}}} \overline{\Delta T}_q^f(x_k,d,\omega_k),$$

to obtain the maximizer d_k and the corresponding Taylor increment $\overline{\Delta T}_q^f(x_k,d_k,\omega_k)$. Compute

$$\texttt{flag} = \text{VERIFY}\Big(\delta_{k-1}, \overline{\Delta T}_q^f(x_k,d_k,\omega_k), \{\epsilon_j\}_{j=1}^q, \omega_k, \tfrac{1}{2}\omega_k\epsilon\Big).$$

Step 1.4: Terminate the ARpDA algorithm with the approximate solution $x_\epsilon = x_k$ if $\texttt{flag} = 1$, or if $\texttt{flag} = 3$, or if $\texttt{flag} = 2$ and (2.15) holds with $\delta = \delta_{k-1}$. Also go to Step 2 of the ARpDA algorithm if $\texttt{flag} = 2$ but (2.15) fails.

Step 1.5: Otherwise (i.e. if $\texttt{flag} = 0$), set

$$\varepsilon_{j,i_\varepsilon+1} = \gamma_\varepsilon \varepsilon_{j,i_\varepsilon} \quad \text{for} \quad j \in \{1,\ldots,p\}, \tag{3.12}$$

increment i_ε by one and return to Step 1.1.

in Step 1.3, the VERIFY algorithm returns `flag` = 1 and thus that $\overline{\Delta T}_q^f(x_k, d_k, \omega_k) = 0$. This means that x_k is a global minimizer of $\overline{T}_q^f(x_k, d, \omega_k)$ in the intersection of a ball of radius δ_{k-1} and $\mathcal{F}$ and $\overline{\Delta T}_q^f(x_k, d, \omega_k) \leq 0$ for any d in this intersection. Thus, for any such d, we obtain from (3.7) with $\xi = \frac{1}{2}\omega_k\epsilon$ that

$$\Delta T_q^f(x_k, d) \leq \overline{\Delta T}_q^f(x_k, d, \omega_k) + \left|\overline{\Delta T}_q^f(x_k, d, \omega_k) - \Delta T_q^f(x_k, d)\right| \leq \tfrac{1}{2}\omega_k\epsilon\chi_q(\delta_{k-1}),$$

which, since $\omega_k \leq 1$, implies (3.14). Suppose next that the VERIFY algorithm returns `flag` = 3. Then $\overline{\Delta T}_q^f(x_k, d_k, \omega_k) > 0$ and thus $d_k \neq 0$. Using the fact that the nature of Step 1.3 ensures that $\overline{\Delta T}_q^f(x_k, d, \omega_k) \leq \overline{\Delta T}_q^f(x_k, d_k, \omega_k)$ for d with $\|d\| \leq \delta_{k-1}$ we have, using (3.9) with $\xi = \frac{1}{2}\omega_k\epsilon$, that, for all such d,

$$\begin{aligned}\Delta T_q^f(x_k, d) &\leq \overline{\Delta T}_q^f(x_k, d, \omega_k) + \left|\overline{\Delta T}_q^f(x_k, d, \omega_k) - \Delta T_q^f(x_k, d)\right| \\ &\leq \overline{\Delta T}_q^f(x_k, d_k, \omega_k) + \left|\overline{\Delta T}_q^f(x_k, d, \omega_k) - \Delta T_q^f(x_k, d)\right| \\ &\leq \epsilon\chi_q(\delta_{k-1})\end{aligned}$$

yielding (3.14). If the VERIFY algorithm returns `flag = 2`, then, for any d with $\|d\| \leq \delta_{k-1}$,

$$\Delta T_q^f(x_k, d) \leq \overline{\Delta T}_q^f(x_k, d, \omega_k) + \left|\overline{\Delta T}_q^f(x_k, d, \omega_k) - \Delta T_q^f(x_k, d)\right| \leq (1+\omega_k)\overline{\Delta T}_q^f(x_k, d_k, \omega_k).$$

Thus, for all d with $\|d\| \leq \delta_{k-1}$,

$$\max\left[0, \Delta T_q^f(x_k, d)\right] \leq (1+\omega_k)\max\left[0, \overline{\Delta T}_q^f(x_k, d_k, \omega_k)\right] = (1+\omega_k)\overline{\phi}_{f,q}^{\delta_{k-1}}(x_k, \omega_k). \quad (3.16)$$

But termination implies that (2.15) holds for $\delta = \delta_{k-1}$, and (3.14) follows with this value of δ. Finally, if the ARpDA algorithm does not terminates within Step 1.4 but Algorithm 3.2 terminates, it must be because the VERIFY algorithm returns `flag` = 2. This implies, as above, that (3.16) holds, which is the rightmost part of (3.15). Similarly, for any d with $\|d\| \leq \delta_{k-1}$,

$$\begin{aligned}\Delta T_q^f(x_k, d) &\geq \overline{\Delta T}_q^f(x_k, d, \omega_k) - \left|\overline{\Delta T}_q^f(x_k, d, \omega_k) - \Delta T_q^f(x_k, d)\right| \\ &\geq \overline{\Delta T}_q^f(x_k, d, \omega_k) - \omega_k\overline{\Delta T}_q^f(x_k, d_k, \omega_k).\end{aligned}$$

Hence

$$\begin{aligned}\operatorname*{globmax}_{\substack{x_k+d\in\mathcal{F}\\ \|d\|\leq\delta_{k-1}}} \Delta T_q^f(x_k, d) &\geq \operatorname*{globmax}_{\substack{x_k+d\in\mathcal{F}\\ \|d\|\leq\delta_{k-1}}} \left[\overline{\Delta T}_q^f(x_k, d, \omega_k) - \omega_k\overline{\Delta T}_q^f(x_k, d_k, \omega_k)\right] \\ &= (1-\omega_k)\overline{\Delta T}_q^f(x_k, d_k, \omega_k).\end{aligned}$$

Since $\overline{\Delta T}_q^f(x_k, d_k, \omega_k) > 0$ when the VERIFY algorithm returns `flag` = 2, we then obtain that, for all $\|d\| \leq \delta_{k-1}$,

$$\max\left[0, \operatorname*{globmax}_{\substack{x_k+d\in\mathcal{F}\\ \|d\|\leq\delta_{k-1}}} \Delta T_q^f(x_k, d)\right] \geq \max\left[0, (1-\omega_k)\overline{\Delta T}_q^f(x_k, d_k, \omega_k)\right] = (1-\omega_k)\overline{\phi}_{f,q}^{\delta_{k-1}}(x_k, \omega_k),$$

which is the leftmost part of (3.15).

3.3 Computing s_k

We now consider computing s_k at Step 2 of the ARpDA algorithm. The process is more complicated than for Step 1, as it potentially involves two situations in which one wishes to guarantee a suitable relative error. The first is when minimizing the model

$$m_k(s) = \overline{f}(x_k,\omega_k) - \overline{\Delta T}_p^f(x_k,s,\omega_k) + \frac{\sigma_k}{(p+\beta)!}\|s\|^{p+\beta}$$

or, equivalently, maximizing

$$-m_k(s) = -\overline{f}(x_k,\omega_k) + \overline{\Delta T}_p^f(x_k,s,\omega_k) - \frac{\sigma_k}{(p+\beta)!}\|s\|^{p+\beta}, \tag{3.17}$$

and the second is when globally minimizing the model's Taylor expansion taken at x_k+s_k in a neighbourhood of diameter δ_k. The first of these situations can be handled in a way very similar to that used above for computing $\overline{\phi}_{f,q}^{\delta_{k-1}}(x_k)$ in Step 1: given a set of approximate derivatives, a step s_k is computed such that it satisfies (2.19) and (2.20), the relative error of the associated $\overline{\Delta T}_p^f(x_k,s_k,\omega_k)$ is then evaluated and, if it is insufficient, the accuracy on the derivative approximations improved and the process restarted. If the relative error on $\overline{\Delta T}_p^f(x_k,s_k,\omega_k)$ is satisfactory and the first test of (2.20) fails, it remains to check that the relative error on $\overline{\phi}_{m_k,q}^{\delta_k}(s_k,\omega_k)$ is also satisfactory. Moreover, as in the original ARpDA algorithm, we have to take into account the possibility that minimizing the model might result in a vanishing decrease. The resulting somewhat involved process is formalized in Algorithm 3.3 on the following page.

Observe that, in Step 2.2, $d_k^{m_k}$ and $\overline{\Delta T}_q^{m_k}(s_k,d_k^{m_k},\omega_k)$ result from the computation of $\overline{\phi}_{m_k,q}^{\delta_k}(s_k,\omega_k)$ which is necessary to verify the second part of (2.20). Note also that we have specified, in the call to VERIFY in Step 2.4 of Algorithm 3.3, absolute accuracy values equal to $\{3\epsilon_j\}_{j=1}^q$. This is because this call aims at checking the accuracy of the Taylor expansion of the *model* and the derivatives which are then approximated are not $\{\overline{\nabla_x^j f(x_k)}\}_{j=1}^q$, but $\{\overline{\nabla_d^j T}_q^{m_k}(s_k,0)\}_{j=1}^q$. It is easy to verify that these (approximate) derivatives are given by

$$\overline{\nabla_d^j T}_q^{m_k}(s_k,0) = \sum_{\ell=j}^p \frac{\overline{\nabla_x^\ell f}(x_k)\|s_k\|^{\ell-j}}{(\ell-j)!} + \left[\nabla_s^j \|s\|^{p+\beta}\right]_{s=s_k}, \tag{3.19}$$

where the last term of the right-hand side is exact. This yields the following error bound.

Lemma 3.3 Suppose that $\|s_k\| \le \mu\epsilon^{\frac{1}{p-q+\beta}}$. Then, for all $j\in\{1,\ldots,p\}$,

$$\left|\overline{\nabla_d^j T}_q^{m_k}(s_k,0) - \nabla_d^j T_q^{m_k}(s_k,0)\right| \le 3\,\varepsilon_j. \tag{3.20}$$

Proof. U □

Algorithm 3.3: Modified Step 2 of the ARpDA algorithm

Step 2: Step calculation.

Step 2.0: The iterate x_k, the radius $\delta_{k-1} \in (0,1]$, the constants $\gamma_\varepsilon \in (0,1)$, $\vartheta \in (0,1)$, the counter i_ε and the absolute accuracies $\{\varepsilon_{j,i_\varepsilon}\}_{j=1}^p$ are given.

Step 2.1: If unavailable, compute $\{\overline{\nabla_x^j f}(x_k)\}_{j=1}^p$ satisfying (3.1) with $\varepsilon_j = \varepsilon_{j,i_\varepsilon}$ for $j \in \{1,\ldots,p\}$.

Step 2.2:
- Attempt to compute a step $s_k \neq 0$ with $x_k + s_k \in \mathcal{F}$ such that (2.19) holds.
- If this not possible, set $\texttt{flag}_s = 1$ and go to Step 2.3.
- Otherwise, pursue the approximate minimization of the model $m_k(s)$ for $x_k + s_k \in \mathcal{F}$ in order to satisfy (2.20), yielding a step s_k, a decrease $\overline{\Delta T}_p^f(x_k, s_k, \omega_k)$ and, if the first part of (2.20) fails, the global maximizer $d_k^{m_k}$ of $\overline{\Delta T}_q^{m_k}(s_k, d, \omega_k)$ subject to $\|d\| \leq \delta_k$ and $x_k + s_k + d \in \mathcal{F}$, together with the corresponding Taylor increment $\overline{\Delta T}_q^{m_k}(s_k, d_k^{m_k}, \omega_k)$.
- Compute
$$\texttt{flag}_s = \mathrm{VERIFY}\Big(\|s_k\|, \overline{\Delta T}_p^f(x_k, s_k, \omega_k), \{\epsilon_j\}_{j=1}^p, \omega_k, \tfrac{1}{2}\omega_k\epsilon\Big).$$
If $\texttt{flag}_s = 0$ go to Step 2.5.

Step 2.3: If $\texttt{flag}_s = 1$ or $\texttt{flag}_s = 3$, compute
$$\operatorname*{globmin}_{x_k+s\in\mathcal{F}} m_k(s),$$
to obtain the minimizer s_k, $\overline{\Delta T}_p^f(x_k, s_k, \omega_k)$.
Set $d_k^{m_k} = 0 = \overline{\Delta T}_q^{m_k}(s_k, d_k^{m_k}, \omega_k)$ and compute
$$\texttt{flag}_s = \mathrm{VERIFY}\Big(\|s_k\|, \overline{\Delta T}_p^f(x_k, s_k, \omega_k), \{\epsilon_j\}_{j=1}^p, \omega_k, \tfrac{1}{2}\omega_k\epsilon\Big).$$
If $\texttt{flag}_s = 0$ go to Step 2.5.

Step 2.4: If $\texttt{flag}_s = 1$ or $\texttt{flag}_s = 3$, terminate the ARpDA algorithm with $x_\epsilon = x_k$. Otherwise, if $\|s_k\| \geq \mu\epsilon^{\frac{1}{p-q+\beta}}$ or if $\|s_k\| < \mu\epsilon^{\frac{1}{p-q+\beta}}$ and
$$\texttt{flag}_d = \mathrm{VERIFY}\Big(\delta_k, \overline{\Delta T}_q^{m_k}(s_k, d_k^{m_k}, \omega_k), \{3\epsilon_j\}_{j=1}^q, \omega_k, \frac{\vartheta(1-\kappa_\omega)}{(1+\kappa_\omega)^2}\,\frac{\omega_k\epsilon}{2}\Big) > 0,$$
go to Step 3 of the ARpDA algorithm with the step s_k, the associated $\overline{\Delta T}_p^f(x_k, s_k, \omega_k)$ and δ_k.

Step 2.5: Set (if $\texttt{flag}_s = 0$ or $\texttt{flag}_d = 0$),
$$\varepsilon_{j,i_\epsilon+1} = \gamma_\varepsilon \varepsilon_{j,i_\epsilon} \quad \text{for} \quad j \in \{1,\ldots,p\}, \tag{3.18}$$
increment i_ε by one and go to Step 2.1.

sing the triangle inequality, (3.19), the inequality $\|s_k\| \leq \mu\epsilon^{\frac{1}{p-q+\beta}} \leq \mu$ and (2.11), we have that, for all $j \in \{1, \ldots, p\}$,

$$\left|\overline{\nabla_d^j T}_q^{m_k}(s_k, 0) - \nabla_d^j T_q^{m_k}(s_k, 0)\right| \leq \sum_{\ell=j}^{p} \frac{\varepsilon_j \|s_k\|^{\ell-j}}{(\ell-j)!} \leq \varepsilon_j \sum_{\ell=j}^{p} \frac{\mu^{\ell-j}}{(\ell-j)!} \leq \varepsilon_j(1+\chi_p(\mu))$$

and (3.20) follows since $\chi_p(\mu) \leq 2\mu$.

Again, Algorithm 3.3 must terminate in a finite number of iterations. Indeed, if after finitely many iterations $\texttt{flag}_s = 1$ or $\texttt{flag}_s = 3$ at the start of Step 2.4, the conclusion is obvious. Suppose now that $\texttt{flag}_s = 2$ at all iterations. If $\|s_k\| < \mu\epsilon^{\frac{1}{p-q+\beta}}$ always hold, the first proposition of Lemma 3.1 ensures that $\texttt{flag}_d > 0$ after finitely many decreases in (3.18), also causing termination. Termination might of course occur if $\|s_k\| \geq \mu\epsilon^{\frac{1}{p-q+\beta}}$ before this limit.

The next Lemma characterizes the outcomes of Algorithm 3.3.

Lemma 3.4 Suppose that the modified Step 2 is used in the ARpDA algorithm. If this algorithm terminates within that step, then there exists a $\delta \in (0, 1]$ such that (2.15) holds for $x = x_\varepsilon$ or

$$\phi_{f,p}^{\|s_k\|}(x_k) \leq \epsilon\chi_p(\|s_k\|). \tag{3.21}$$

Otherwise we have that (2.19) and

$$\left|\overline{\Delta T}_p^f(x_k, s_k, \omega_k) - \Delta T_p^f(x_k, s_k)\right| \leq \omega_k \overline{\Delta T}_p^f(x_k, s_k, \omega_k) \tag{3.22}$$

are satisfied. Moreover, either $\|s_k\| \geq \mu\epsilon^{\frac{1}{p-q+\beta}}$, or

$$\phi_{m_k,q}^{\delta_k}(s_k) \quad \leq \quad (1+\kappa_\omega) \max\left[\frac{\vartheta(1-\kappa_\omega)}{(1+\kappa_\omega)^2}\,\epsilon, \frac{\theta\|s_k\|^{p-q+\beta}}{(p-q+\beta)!}\right] \chi_q(\delta_k). \tag{3.23}$$

hold.

Proof. W □

e first note that, because of (3.17) and because Step 2.2 imposes (2.19), we have that $\overline{\Delta T}_p^f(x_k, s_k, \omega_k) \geq 0$ at the end of this step. Let us first consider the case where the calls to the VERIFY algorithm in Step 2.2 and in Step 2.3 both return $\texttt{flag}_s = 1$ or $\texttt{flag}_s = 3$ and note that Step 2.1 yields (3.5) with $T_r = T_r^f$, $r = p$ and $\{\zeta_j\}_{j=1}^r = \{\varepsilon_{j,i_\varepsilon}\}_{j=1}^p$. Moreover, $\omega = \omega_k \in (0, 1)$ so that we can use Lemma 3.1 to analyse the outcome of the above calls to the VERIFY Algorithm. If $\|s_k\| = 0$, Lemma 2.4 ensures that (2.15) holds for $x = x_\varepsilon$ for a radius $\delta \in (0, 1)$. Otherwise, we have that $\overline{\Delta T}_p^f(x_k, s, \omega_k) > 0$ because of (2.19) and, since s_k is then a global minimizer of m_k, that

$$\overline{\Delta T}_p^f(x_k, s, \omega_k) - \frac{\sigma_k}{(p+\beta)!}\|s\|^{p+\beta} \leq \overline{\Delta T}_p^f(x_k, s_k, \omega_k) - \frac{\sigma_k}{(p+\beta)!}\|s_k\|^{p+\beta} \tag{3.24}$$

for all s. Thus, if $\|s\| \leq \|s_k\|$, then $\overline{\Delta T}_p^f(x_k, s, \omega_k) \leq \overline{\Delta T}_p^f(x_k, s_k, \omega_k)$. This implies that

$$\operatorname*{globmax}_{\substack{x_k+s\in\mathcal{F}\\ \|s\|\leq\|s_k\|}} \overline{\Delta T}_p^f(x_k, s, \omega_k) = \overline{\Delta T}_p^f(x_k, s_k, \omega_k).$$

We may now repeat the proof of Lemma 3.2 for the cases $\texttt{flag}_s \in \{1, 3\}$, with q replaced by p and δ_{k-1} replaced by $\|s_k\|$, and deduce that (3.21) holds.

Assume now that Algorithm 3.3 terminates in Step 2.4. This means that the VERIFY algorithm invoked in either Step 2.2 or Step 2.3 terminates with $\texttt{flag}_s = 2$, and we deduce from (3.8) that(3.22) holds.

Let us now consider the case $\|s_k\| < \mu\epsilon^{\frac{1}{p-q+\beta}}$ and note that Lemma 3.3 ensures that (3.5) is satisfied with $T_r = T_r^{m_k}$, $r = q$ and $\{\zeta_j\}_{j=1}^r = \{3\varepsilon_{j,i_\varepsilon}\}_{j=1}^q$. Moreover, the triangle inequality gives

$$\Delta T_q^{m_k}(s_k, d) \leq \overline{\Delta T}_q^{m_k}(s_k, d, \omega_k) + \left|\overline{\Delta T}_q^{m_k}(s_k, d, \omega_k) - \Delta T_q^{m_k}(s_k, d)\right|. \tag{3.25}$$

First, assume that, in Step 2.4, Algorithm 3.3 terminates because $\texttt{flag}_d = 1$ is returned by VERIFY. Then, $\overline{\Delta T}_q^{m_k}(s_k, d_k^{m_k}, \omega_k) = 0$. Moreover, using (3.25), the definition of $d_k^{m_k}$ given at Step 2.2 of Algorithm 3.3, (3.7) and recalling that $\omega_k \leq 1$, we obtain that, for all d with $\|d\| \leq \delta_k$,

$$\begin{aligned}
\Delta T_q^{m_k}(s_k, d) &\leq \overline{\Delta T}_q^{m_k}(s_k, d, \omega_k) + \left|\overline{\Delta T}_q^{m_k}(s_k, d, \omega_k) - \Delta T_q^{m_k}(s_k, d)\right| \\
&\leq \overline{\Delta T}_q^{m_k}(s_k, d_k^{m_k}, \omega_k) + \left|\overline{\Delta T}_q^{m_k}(s_k, d, \omega_k) - \Delta T_q^{m_k}(s_k, d)\right| \\
&= \left|\overline{\Delta T}_q^{m_k}(s_k, d, \omega_k) - \Delta T_q^{m_k}(s_k, d)\right| \\
&\leq \frac{\vartheta(1-\kappa_\omega)}{2(1+\kappa_\omega)^2}\,\omega_k\,\epsilon\,\chi_q(\|d\|) \\
&\leq \frac{\vartheta(1-\kappa_\omega)}{(1+\kappa_\omega)^2}\,\epsilon\,\chi_q(\delta_k).
\end{aligned} \tag{3.26}$$

If, instead, termination occurs with VERIFY returning $\texttt{flag}_d = 2$, then we will show that for all d with $\|d\| \leq \delta_k$,

$$\Delta T_q^{m_k}(s_k, d) \leq (1+\omega_k)\overline{\Delta T}_q^{m_k}(s_k, d_k^{m_k}, \omega_k) \leq (1+\omega_k)\frac{\theta\|s_k\|^{p-q+\beta}}{(p-q+\beta)!}\,\chi_q(\delta_k). \tag{3.27}$$

Indeed, from (3.25), (3.8), (2.27) and the definition of $d_k^{m_k}$ at Step 2.2 of Algorithm 3.3, we obtain for all d with $\|d\| \leq \delta_k$

$$\begin{aligned}
\Delta T_q^{m_k}(s_k, d) &\leq (1+\omega_k)\overline{\Delta T}_q^{m_k}(s_k, d_k^{m_k}, \omega_k), \\
&\leq (1+\omega_k)\max\left[0, \operatorname*{globmax}_{\substack{x_k+s_k+d\in\mathcal{F}\\ \|d\|\leq\delta_k}} \overline{\Delta T}_q^{m_k}(s_k, d, \omega_k)\right] \\
&= (1+\omega_k)\overline{\phi}_{m_k,q}^{\delta_k}(s_k, \omega_k),
\end{aligned}$$

in which the equality follows from the definition (2.14). We can then conclude, using (2.20), that (3.27) holds for all d with $\|d\| \leq \delta_k$.

Finally, if termination occurs instead because VERIFY returns $\texttt{flag}_d = 3$, we deduce from the (3.25) and (3.9) that, for all d with $\|d\| \leq \delta_k$,

$$\Delta T_q^{m_k}(s_k, d) \leq \frac{\vartheta(1-\kappa_\omega)}{(1+\kappa_\omega)^2}\,\epsilon\,\chi_q(\delta_k) \tag{3.28}$$

Observe now that (2.27), (2.7) (for m_k at s_k) and each of (3.26), (3.27) or (3.28) ensures (3.23).

Note that (3.21) can be viewed as a stronger optimality condition than (2.10) since it implies that the p-th (rather than q-th with $q \leq p$) order Taylor expansion of f around x_k is bounded below by a correctly scaled multiple of ϵ, and in a possibly larger neighbourhood. It is thus acceptable to terminate the ARpDA algorithm with $x_\epsilon = x_k$ as stated in Step 2.4 of Algorithm 3.3.

3.4 The complexity of a single ARpDA iteration

The last part of this section is devoted to bounding the evaluation complexity of a single iteration of the ARpDA algorithm. The count in (approximate) objective function evaluations is the simplest: these only occur in Step 3 which requires at most two such evaluations.

Now observe that evaluations of $\{\overline{\nabla_x^j f}\}_{j=1}^p$ possibly occur in Steps 1.2 and 2.1. However it is important to note that, within these steps, the *derivatives are evaluated only if the current values of the absolute errors are smaller than that used for the previous evalutions* of the same derivative at the same point (x_k). Moreover, these absolute errors are, by construction, linearly decreasing with rate γ_ε within the same iteration of the ARpDA algorithm (they are initialized in Step 1.1, decreased each time by a factor γ_ε in (3.12) invoked in Step 1.5, down to values $\{\varepsilon_{j,i_\varepsilon}\}_{j=1}^p$ which are then passed to the modified Step 2, and decreased there further in (3.18) in Step 2.5, again by successive multiplication with γ_ε). Furthermore, we have argued already, both for the modified Step 1 and the modified Step 2, that any of these algorithms terminates as soon as (3.6) holds for the relevant value of ξ, which we therefore need to determine. For Step 1, this value is $\frac{1}{2}\omega_k\epsilon$, while, for Step 2, it is given by

$$\min\left[\tfrac{1}{2}\omega_k\epsilon, \frac{\vartheta(1-\kappa_\omega)}{(1+\kappa_\omega)^2}\frac{\omega_k\epsilon}{2}\right] = \frac{\vartheta(1-\kappa_\omega)}{2(1+\kappa_\omega)^2}\,\omega_k\epsilon \tag{3.29}$$

when $\|s_k\| < \mu\epsilon^{\frac{1}{p-q+\beta}}$ and by $\frac{1}{2}\omega_k\epsilon$ when $\|s_k\| \geq \mu\epsilon^{\frac{1}{p-q+\beta}}$. As a consequence, we obtain the following lemma.

Lemma 3.5 Suppose that $\omega_k \geq \omega_{\min} > 0$ for all k. Then each iteration of the ARpDA algorithm involves at most 2 (approximate) evaluations of the objective function and at most $1 + \nu_{\max}(\epsilon)$ (approximate) evaluations of its p first derivatives, where

$$\nu_{\max}(\epsilon) = \left\lfloor \frac{1}{\log(\gamma_\varepsilon)} \left\{ \log\left(\frac{\vartheta(1-\kappa_\omega)}{6(1+\kappa_\omega)^2}\omega_{\min}\epsilon \right) - \log(\kappa_\varepsilon) \right\} \right\rfloor. \tag{3.30}$$

Proof. T □

he upper bound on the (approximate) function evaluations immediately follows from the observation that, as mentioned at the beginning of the current paragraph, these computations occur at most twice in Step 3 of Algorithm 2.1. Concerning the second part of the thesis we notice that, from Lemma 3.3, in Step 2.4 of Algorithm 3.3 we have to make $\{\overline{\nabla_x^j f}(x_k)\}_{j=1}^p$ three times more accurate than the desired accuracy in $\{\overline{\nabla_d^j T}_q^{m_k}(s_k,0)\}_{j=1}^q$, when $\|s_k\| < \mu\epsilon^{\frac{1}{p-q+\beta}}$ (the input values for the absolute accuracy values in the VERIFY call are $\{3\epsilon_j\}_{j=1}^q$). Thus, the VERIFY Algorithm stops whenever

$$\max_{j\in\{1,\ldots,q\}} \varepsilon_j \leq \frac{\vartheta(1-\kappa_\omega)\omega_k\epsilon}{6(1+\kappa_\omega)^2}.$$

We may thus conclude from Lemma 3.1 that no further reduction in $\{\varepsilon_j\}_{j=1}^p$ (and hence no further approximation of $\{\overline{\nabla_x^j f}(x_k)\}_{j=1}^p$) will occur once i_ε, the number of decreases in $\{\varepsilon_j\}_{j=1}^p$, is large enough to ensure that

$$\gamma_\varepsilon^{i_\epsilon}[\max_{j\in\{1,\ldots,p\}} \varepsilon_{j,0}] \leq \frac{\vartheta(1-\kappa_\omega)}{6(1+\kappa_\omega)^2}\omega_{\min}\epsilon$$

(Note that this inequality could hold for $i_\epsilon = 0$.) Because of our assumption that $\omega_k \geq \omega_{\min}$ and (3.11), the above inequality is then verified when

$$i_\epsilon \leq \left\lfloor \frac{1}{\log(\gamma_\varepsilon)} \left\{ \log\left(\frac{\vartheta(1-\kappa_\omega)}{6(1+\kappa_\omega)^2}\omega_{\min}\epsilon \right) - \log(\kappa_\varepsilon) \right\} \right\rfloor,$$

which concludes the proof when taking into account that the derivatives must be computed at least once per iteration.

Note that, for simplicity, we have ignored the fact that only $q \leq p$ derivatives need to be evaluated in Steps 1.2. Lemma 3.5 can obviously be refined to reflect this observation.

We conclude this section by a comment on what happens whenever exact objective function and derivatives are used. In that case the (exact) derivatives are computed only once per iteration of the ARpDA algorithm (in Step 1.2 for the first q and in Step 2.1 for the remaining $p-q$) and every other call to VERIFY returns $\texttt{flag} = 1$ or $\texttt{flag} = 2$. Moreover, there is no need to recompute $\overline{f}$ to obtain (2.22) in Step 3. The evaluation complexity of a single iteration of the ARpDA algorithm then reduces to a single evaluation of f and its first p derivatives (and $\nu_{\max}(\epsilon) = 1$ for all k), as expected.

4 Evaluation complexity of the deterministic ARpDA

This section is devoted to the evaluation complexity analysis of the ARpDA algorithm in the deterministic context. We start by providing a simple lower bound on the model decrease.

Lemma 4.1 [13, Lemma 3.1] The mechanism of the ARpDA algorithm guarantees that, for all $k \geq 0$,

$$\overline{\Delta T}_p^f(x_k, s_k, \omega_k) > \frac{\sigma_k}{(p+\beta)!}\|s_k\|^{p+\beta}, \tag{4.1}$$

and so (2.23) is well-defined.

Proof. W □

e have that

$$0 < m_k(0) - m_k(s_k) = \overline{T}_p(x_k, 0, \omega_k) - \overline{T}_p(x_k, s_k, \omega_k) - \frac{\sigma_k}{(p+\beta)!}\|s_k\|^{p+\beta}.$$

We next show that the regularization parameter σ_k has to remain bounded, even in the presence of inexact computation of f and its derivatives. This lemma hinges heavily on (2.13), (2.21) and (2.22).

Lemma 4.2 Let $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n)$. Then, for all $k \geq 0$,

$$\sigma_k \leq \sigma_{\max} \stackrel{\rm def}{=} \max\left[\sigma_0, \frac{\gamma_3(L+3)}{1-\eta_2}\right] \tag{4.2}$$

and

$$\omega_k \geq \omega_{\min} \stackrel{\rm def}{=} \min\left[\kappa_\omega, \frac{1}{\sigma_{\max}}\right]. \tag{4.3}$$

Proof. A □

ssume that

$$\sigma_k \geq \frac{L+3}{1-\eta_2}. \tag{4.4}$$

Also observe that, because of the triangle inequality, (3.22) (as ensured by Lemma 3.4) and (2.22),

$$\begin{aligned} |\overline{T}_p^f(x_k, s_k, \omega_k) - T_p^f(x_k, s_k)| &\leq |\overline{f}_k(x_k, \omega_k) - f(x_k)| \\ &\quad + |\overline{\Delta T}_p^f(x_k, s_k, \omega_k) - \Delta T_p^f(x_k, s_k)| \\ &\leq 2\omega_k |\overline{\Delta T}_p^f(x_k, s_k, \omega_k)| \end{aligned}$$

and hence, again using the triangle inequality, (2.21), (2.5), (2.25), (4.1) and (4.4),

$$\begin{aligned} |\rho_k - 1| &\leq \frac{|\overline{f}_k(x_k+s_k, \omega_k) - \overline{T}_p^f(x_k, s_k, \omega_k)|}{\overline{\Delta T}_p^f(x_k, s_k, \omega_k)} \\ &\leq \frac{1}{\overline{\Delta T}_p^f(x_k, s_k, \omega_k)}\Big[|\overline{f}_k(x_k+s_k, \omega_k) - f(x_k+s_k)| + |f(x_k+s_k) - T_p^f(x_k, s_k)| \\ &\qquad + |\overline{T}_p^f(x_k, s_k, \omega_k) - T_p^f(x_k, s_k)|\Big] \\ &\leq \frac{1}{\overline{\Delta T}_p^f(x_k, s_k, \omega_k)}\Big[|f(x_k+s_k) - T_p^f(x_k, s_k)| + 3\omega_k \overline{\Delta T}_p^f(x_k, s_k, \omega_k)\Big] \\ &\leq \frac{1}{\overline{\Delta T}_p^f(x_k, s_k, \omega_k)}\Big[\frac{L}{(p+\beta)!}\|s_k\|^{p+\beta} + \frac{3\overline{\Delta T}_p^f(x_k, s_k, \omega_k)}{\sigma_k}\Big] \\ &< \frac{L}{\sigma_k} + \frac{3}{\sigma_k} \\ &\leq 1 - \eta_2 \end{aligned}$$

and thus that $\rho_k \geq \eta_2$. Then iteration k is very successful in that $\rho_k \geq \eta_2$ and, because of (2.24), $\sigma_{k+1} \leq \sigma_k$. As a consequence, the mechanism of the algorithm ensures that (4.2) holds. Observe now that this result and (2.25) imply that, for all k, ω_k may be chosen such that $\min[\kappa_\omega, \sigma_{\max}^{-1}] \leq \omega_k \leq \kappa_\omega$, yielding (4.3).
It is important to note that (4.3) in this lemma provides the lower bound on ω_k required in Lemma 3.5. We now borrow a technical result from [13].

Lemma 4.3 [13, Lemma 2.4] Let s be a vector of $\mathbb{R}^n$ and $p \in \mathbb{N}_0$ and $\beta \in (0,1]$ such that $j \in \{0, \ldots, p\}$. Then

$$\| \nabla_s^j (\|s\|^{p+\beta}) \|_{[j]} \leq \frac{(p+\beta)!}{(p-j+\beta)!} \|s\|^{p-j+\beta}. \tag{4.5}$$

Our next move is to prove a lower bound on the step norm. While the proof of this result is clearly inspired from that of [13, Lemma 3.3], it nevertheless crucially differs when approximate values are considered instead of exact ones.

Lemma 4.4 Let $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n)$. Then, for all $k \geq 0$ such that the ARpDA algorithm does not terminate at iteration $k+1$,

$$\|s_k\| \geq \kappa_s \epsilon^{\frac{1}{p-q+\beta}}, \tag{4.6}$$

where

$$\kappa_s \stackrel{\rm def}{=} \min\left\{ \mu, \left[\frac{(1-\kappa_\omega)(1-\vartheta)(p-q+\beta)!}{(1+\kappa_\omega)(L+\sigma_{\max}+\theta(1+\kappa_\omega))} \right]^{\frac{1}{p-q+\beta}} \right\}. \tag{4.7}$$

Proof. I □

f $\|s_k\| \geq \mu \epsilon^{\frac{1}{p-q+\beta}}$, the result is obvious. Suppose now that

$$\|s_k\| < \mu \epsilon^{\frac{1}{p-q+\beta}}. \tag{4.8}$$

Since the algorithm does not terminate at iteration $k+1$, we have that

$$\overline{\phi}_{f,q}^{\delta_k}(x_{k+1}) > \frac{\epsilon}{1+\omega_k} \chi_q(\delta_k)$$

and therefore, using (3.15), that

$$\phi_{f,q}^{\delta_k}(x_{k+1}) > \frac{1-\omega_k}{1+\omega_k} \epsilon \, \chi_q(\delta_k). \tag{4.9}$$

Let the global minimum in the definition of $\phi_{f,q}^{\delta_k}(x_{k+1})$ be achieved at d with $\|d\| \leq \delta_k$. Then,

using (2.7), the triangle inequality and (4.5), we deduce that

$$\begin{aligned}
\phi_{f,q}^{\delta_k}(x_{k+1}) &= -\sum_{\ell=1}^{q} \frac{1}{\ell!}\nabla_x^\ell f(x_{k+1})[d]^\ell \\
&\le \left|\sum_{\ell=1}^{q} \frac{1}{\ell!}\nabla_x^\ell f(x_{k+1})[d]^\ell - \sum_{\ell=1}^{q} \frac{1}{\ell!}\nabla_s^\ell T_p^f(x_k,s_k)[d]^\ell\right| - \sum_{\ell=1}^{q} \frac{1}{\ell!}\nabla_s^\ell T_p^f(x_k,s_k)[d]^\ell \\
&\le \sum_{\ell=1}^{q} \frac{1}{\ell!}\Big[\|\nabla_x^\ell f(x_{k+1}) - \nabla_s^\ell T_p^f(x_k,s_k)\|_{[\ell]}\Big]\delta_k^\ell \\
&\qquad\qquad - \sum_{\ell=1}^{q} \frac{1}{\ell!}\left(\nabla_s^\ell\left[T_p^f(x_k,s) + \frac{\sigma_k}{(p+\beta)!}\|s\|^{p+\beta}\right]_{s=s_k}\right)[d]^\ell \\
&\qquad\qquad + \sum_{\ell=1}^{q} \frac{\sigma_k}{\ell!(p-\ell+\beta)!}\|s_k\|^{p-\ell+\beta}\delta_k^\ell. \qquad (4.10)
\end{aligned}$$

Now, because of (2.16), (2.7) (for m_k at s_k) and the fact that $\|d\| \le \delta_k$, we have that

$$-\sum_{\ell=1}^{q} \frac{1}{\ell!}\left(\nabla_s^\ell\left[T_p^f(x_k,s) + \frac{\sigma_k}{(p+\beta)!}\|s\|^{p+\beta}\right]_{s=s_k}\right)[d]^\ell = \Delta T_q^{m_k}(s_k,d) \le \phi_{m_k,q}^{\delta_k}(s_k).$$

Then, as $\|s_k\| < \mu\epsilon^{\frac{1}{p-q+\beta}} < 1$ because of (4.8), we may use (3.23) (ensured by Lemma 3.4) and (2.6) and distinguish the cases where the maximum in (3.23) is attained in its first or its second argument. In the latter case, we deduce from (4.10) that

$$\begin{aligned}
\phi_{f,q}^{\delta_k}(x_{k+1}) &\le \sum_{\ell=1}^{q} \frac{L}{\ell!(p-\ell+\beta)!}\|s_k\|^{p-\ell+\beta}\delta_k^\ell + (1+\kappa_\omega)\frac{\theta\,\chi_q(\delta_k)}{(p-q+\beta)!}\|s_k\|^{p-q+\beta} \\
&\qquad + \sum_{\ell=1}^{q} \frac{\sigma_k}{\ell!(p-\ell+\beta)!}\|s_k\|^{p-\ell+\beta}\delta_k^\ell \\
&\le \frac{\Big[L+\sigma_k+\theta(1+\kappa_\omega)\Big]\chi_q(\delta_k)}{(p-q+\beta)!}\|s_k\|^{p-q+\beta}; \qquad (4.11)
\end{aligned}$$

otherwise, (4.10) guarantees that

$$\phi_{f,q}^{\delta_k}(x_{k+1}) \le \frac{(L+\sigma_k)\chi_q(\delta_k)}{(p-q+\beta)!}\|s_k\|^{p-q+\beta} + \frac{\vartheta(1-\kappa_\omega)}{1+\kappa_\omega}\,\epsilon\chi_q(\delta_k). \qquad (4.12)$$

Using now (4.9), (2.27), (4.8), (4.11) and (4.12),we thus have that

$$\begin{aligned}
\|s_k\| &\ge \min\left\{\mu\epsilon^{\frac{1}{p-q+\beta}}, \left[\frac{\epsilon(1-\kappa_\omega)(p-q+\beta)!}{(1+\kappa_\omega)(L+\sigma_k+\theta(1+\kappa_\omega))}\right]^{\frac{1}{p-q+\beta}}, \left[\frac{\epsilon(1-\kappa_\omega)(1-\vartheta)(p-q+\beta)!}{(1+\kappa_\omega)(L+\sigma_k)}\right]^{\frac{1}{p-q+\beta}}\right\} \\
&\ge \min\left\{\mu\epsilon^{\frac{1}{p-q+\beta}}, \left[\frac{\epsilon(1-\kappa_\omega)(1-\vartheta)(p-q+\beta)!}{(1+\kappa_\omega)(L+\sigma_k+\theta(1+\kappa_\omega))}\right]^{\frac{1}{p-q+\beta}}\right\},
\end{aligned}$$

where we have used the fact that $\theta \in (0,1)$ to obtain the last inequality. Then (4.6) follows from (4.2).

We now combine all the above results to deduce an upper bound on the maximum number of successful iterations, from which a final complexity bound immediately follows.

Theorem 4.5 Let $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n)$ and $\epsilon \in (0,1)$ be given. Then the ARpDA algorithm using the modified Steps 1 (on page 13) and 2 (on page 16) produces an iterate x_ϵ such that (2.10) or (3.21) holds in at most

$$\left\lfloor \kappa_p(f(x_0)-f_{\rm low})\left(\epsilon^{-\frac{p+\beta}{p-q+\beta}}\right)\right\rfloor + 1 \tag{4.13}$$

successful iterations,

$$\tau(\epsilon) \stackrel{\rm def}{=} \left\lfloor \left\{ \left\lfloor \kappa_p(f(x_0)-f_{\rm low})\left(\epsilon^{-\frac{p+\beta}{p-q+\beta}}\right)+1\right\rfloor \left(1+\frac{|\log\gamma_1|}{\log\gamma_2}\right)+\frac{1}{\log\gamma_2}\log\left(\frac{\sigma_{\max}}{\sigma_0}\right)\right\}\right\rfloor \tag{4.14}$$

iterations in total, $2\tau(\epsilon)$ (approximate) evaluations of f and $(1+\nu_{\max}(\epsilon))\tau(\epsilon)$ approximate evaluations of $\{\nabla_x^j f\}_{j=1}^p$, where $\sigma_{\max}$ is given by (4.2), $\omega_{\min}$ by (4.3), $\nu_{\max}(\epsilon)$ by (3.30), and where

$$\kappa_p \stackrel{\rm def}{=} \frac{(p+\beta)!}{\eta_1(1-\alpha)\sigma_{\min}}\max\left\{\frac{1}{\mu^{p+\beta}},\left[\frac{(1+\kappa_\omega)(L+\sigma_{\max}+\theta(1+\kappa_\omega))}{(1-\kappa_\omega)(1-\vartheta)(p-q+\beta)!}\right]^{\frac{p+\beta}{p-q+\beta}}\right\}. \tag{4.15}$$

Proof. A □

t each successful iteration k before termination the algorithm guarantees the decrease

$$\begin{aligned} f(x_k)-f(x_{k+1}) &\geq [\overline{f}_k(x_k,\omega_k)-\overline{f}_k(x_{k+1},\omega_k)] - 2\omega_k\overline{\Delta T}_p^f(x_k,s_k,\omega_k) \\ &\geq \eta_1\overline{\Delta T}_p^f(x_k,s_k,\omega_k) - \alpha\eta_1\overline{\Delta T}_p^f(x_k,s_k,\omega_k) \\ &> \frac{\eta_1(1-\alpha)\sigma_{\min}}{(p+\beta)!}\,\|s_k\|^{p+\beta}, \end{aligned} \tag{4.16}$$

where we used (2.18), (2.21), (2.22), (2.23), (4.1) and (2.24). Moreover we deduce from (4.16) and (4.6) that

$$f(x_k)-f(x_{k+1}) \geq \kappa_p^{-1}\epsilon^{\frac{p+\beta}{p-q+\beta}} \;\text{ where }\; \kappa_p^{-1} \stackrel{\rm def}{=} \frac{\eta_1(1-\alpha)\sigma_{\min}\kappa_s^{p+\beta}}{(p+\beta)!}. \tag{4.17}$$

Thus, since $\{f(x_k)\}$ decreases monotonically,

$$f(x_0)-f(x_{k+1}) \geq \kappa_p^{-1}\,\epsilon^{\frac{p+\beta}{p-q+\beta}}\,|\mathcal{S}_k|.$$

Using that f is bounded below by $f_{\rm low}$, we conclude that

$$|\mathcal{S}_k| \leq \kappa_p(f(x_0)-f_{\rm low})\epsilon^{-\frac{p+\beta}{p-q+\beta}} \tag{4.18}$$

until termination, and the desired bound on the number of successful iterations follows. Lemma 2.2 is then invoked to compute the upper bound on the total number of iterations, and Lemma 3.5 to deduce the upper bounds on the number of evaluations of f and its derivatives.

We emphasize that (4.13) was shown in [13] to be optimal for a quite wide class of minimization algorithms. The slightly weaker bound $(1+\nu_{\max}(\epsilon))\tau(\epsilon)$ may be seen as the (very modest) price to pay for allowing inexact evaluations.

Focusing on the order in ϵ and using (4.14), we therefore obtain the following condensed result on evaluation complexity for nonconvex optimization.

Theorem 4.6 Let $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n)$. Then, given $\epsilon \in (0,1)$, the ARpDA algorithm using the modified Steps 1 (on page 13) and 2 (on page 16) needs at most

$$O\left(\epsilon^{-\frac{p+\beta}{p-q+\beta}}\right) \text{ iterations and (approximate) evaluations of } f$$

and at most

$$O\left(|\log(\epsilon)|\epsilon^{-\frac{p+\beta}{p-q+\beta}}\right) \text{ (approximate) evaluations of the } p \text{ first derivatives}$$

to compute an (ϵ,δ)-approximate q-th-order-necessary minimizer for the set-constrained problem (2.1).

In particular, if the p-th derivative of f is assumed to be globally Lipschitz rather than merely Hölder continuous (i.e. if $\beta = 1$), these orders reduce to

$$O\left(\epsilon^{-\frac{p+1}{p-q+1}}\right) \text{ iterations and (approximate) evaluations of } f$$

and at most

$$O\left(|\log(\epsilon)|\epsilon^{-\frac{p+1}{p-q+1}}\right) \text{ (approximate) evaluations of the } p \text{ first derivatives.}$$

As indicated in the comment at the end of Section 3, all $O(|\log(\epsilon)|)$ terms reduce to a constant independent of ϵ if exact evaluations of f and its derivatives are used, and the above results then recover the optimal complexity results of [13].

We conclude this section by commenting on the special case where the objective function evaluations are exact and that of the derivatives inexact. We first note that this case is already covered by the theory presented above (since (2.21) and (2.22) automatically holds as their left-hand side is identically zero), but this remark also shows that the ARpDA algorithm can be simplified by replacing the computation of $\overline{f}(x_k+s_k,\omega_k)$ by that of $f(x_k+s_k)$ and by skipping the verification and possible recomputation of $\overline{f}(x_k,\omega_k)$ entirely. As consequence, the ARpDA algorithm only evaluates the exact objective function f *once* per iteration, and the maximum number of such evaluations is therefore given by $\tau(\epsilon)$ instead of $2\tau(\epsilon)$, while the maximum number of (inexact) derivatives evaluations is still given by $(1+\nu_{\max}(\epsilon))\tau(\epsilon)$.

5 A variant of the ARpDA algorithm

We now describe a variant of the ARpDA algorithm for which an even better complexity can be proved, but at the price of a more restrictive dynamic accuracy strategy. In the Step 1.0 of the ARpDA algorithm, we allow the choice of an arbitrary set of $\{\varepsilon_{j,0}\}_{j=1}^{p}$ with the constraint that $\varepsilon_{j,0} \leq \kappa_\varepsilon$ for $j \in \{1,\ldots,p\}$. This allows these accuracy thresholds to vary non-monotonically from iteration to iteration, providing considerable flexibility and allowing

large inaccuracies even if these thresholds were made small in past iterations due to local nonlinearity. A different, more rigid, strategy is also possible: suppose that the thresholds $\{\varepsilon_{j,0}\}_{j=1}^{p}$ are not reset at each iteration, that is

$$\text{Step 1.1 is only executed for } k = 0. \tag{5.1}$$

This clearly results in a monotonic decrease of each ε_j across all iterations. As a consequence, $\nu_{\max}(\epsilon)$ in (3.30) now bounds the total number of reductions of the ε_j over all iterations, i.e. on the *entire run of the algorithm*. We then deduce that the total number of derivatives evaluation is then bounded by $\nu_{\max}(\epsilon) + \tau(\epsilon)$ (instead of $(1 + \nu_{\max}(\epsilon))\tau(\epsilon)$) and we may establish the worst-case complexity of the resulting "monotonic" variant as follows.

Theorem 5.1 Let $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n)$ and $\epsilon \in (0,1)$ be given. Then the ARpDA algorithm using the modified Steps 1 (on page 13) and 2 (on page 16) as well as the modified rule (5.1) produces an iterate x_ϵ such that (2.10) or (3.21) holds in at most (4.13) successful iterations, $\tau(\epsilon)$ iterations in total, $2\tau(\epsilon)$ (approximate) evaluations of f and $\nu_{\max}(\epsilon)+\tau(\epsilon)$ approximate evaluations of $\{\nabla_x^j f\}_{j=1}^{p}$, where $\tau(\epsilon)$ is given by (4.14), κ_p is given by (4.15), $\sigma_{\max}$ by (4.2), $\omega_{\min}$ by (4.3) and $\nu_{\max}(\epsilon)$ by (3.30).

As above, this complexity bound can be condensed to

$$\begin{array}{l} O\left(\epsilon^{-\frac{p+\beta}{p-q+\beta}}\right) \text{ iterations and (approximate) evaluations of } f \\ O\left(|\log(\epsilon)| + \epsilon^{-\frac{p+\beta}{p-q+\beta}}\right) \text{ (approximate) evaluations of the } p \text{ first derivatives,} \end{array} \tag{5.2}$$

typically improving on Theorem 4.6. When $p = 2$, $q = 1$ and $\beta = 1$, the ARpDA variant using the more restrictive accuracy strategy (5.1) requires at most

$$O\Big(|\log(\epsilon)| + \epsilon^{-3/2}\Big)$$

(approximate) evaluations of the gradient, which corresponds to the bound derived for the ARC-DFO algorithm of [12]. This is not surprising as this latter algorithm uses a monotonically decreasing sequence of finite-difference stepsizes, implying monotonically decreasing gradient-accuracy thresholds.

One should however notice that the improved bound (5.2) comes at the price of asking, for potentially many iterations, an accuracy on $\{\nabla_x^j f\}_{j=1}^{p}$ which is tighter than what is needed to ensure progress of the minimization. In practice, this might be a significant drawback. We will thus restrict our attention, in what follows, to the original ARpDA algorithm, but similar developments are obviously possible for the "monotonic" variant just discussed.

6 Application to unconstrained and bound-constrained first- and second-order nonconvex inexact optimization

Because of its wide-ranging applicability, the framework discussed above may appear somewhat daunting in its generality. Moreover, the fact that it involves (possibly constrained)

global optimization subproblems in several of its steps may suggest that it has to remain conceptual. We show in this section that *this is not the case*, and stress that it is *much simpler when specialized to small values of p and q* (which are, for now, the most practical ones) and that our approach leads to elegant and implementable numerical algorithms. To illustrate this point, we now review what happens for $p \leq 2$.

We first discuss the case where one seeks to compute a first-order critical point for an unconstrained optimization problem using approximate function values as well as approximate first derivatives. For simplicity of exposition, we will also assume that the gradient of f is Lipschitz (rather than Hölder) continuous. In our general context, this means that we consider the case where $q = 1$, $p = 1$, $\beta = 1$ and $\mathcal{F} = \mathbb{R}^n$. We first note that, as pointed out in (2.8),

$$\phi^{\delta}_{f,1}(x) = \|\nabla^1_x f(x)\|\delta \quad \text{and} \quad \overline{\phi}^{\delta}_{f,1} = \|\overline{\nabla^1_x f}(x)\|\delta \quad \text{irrespective of} \quad \delta \in (0,1], \tag{6.1}$$

which means that, since we can choose $\delta = 1$, Step 1 of the ARpdA algorithm reduces to the computation of an approximate gradient $\overline{\nabla^1_x f}(x_k)$ with relative error ω_k and verification that ϵ-approximate optimality is not yet achieved. If that is the case, computing s_k at Step 2 is also extremely simple since it is easy to verify that

$$s_k = s^*_k = -\frac{1}{\sigma_k}\overline{\nabla^1_x f}(x_k).$$

Lemma 2.3 then ensures that this step is acceptable for some $\delta_k \in (0,1]$, the value of which being irrelevant since it is not used in Step 1 of the next iteration. Moreover, if the relative error on $\overline{\nabla^1_x f}(x_k)$ is bounded by ω_k, then

$$\begin{aligned} |\overline{\Delta T}^f_1(x_k,s_k) - \Delta T^f_1(x_k,s_k)| &\leq \|\overline{\nabla^1_x f}(x_k) - \nabla^1_x f(x_k)\| \frac{\|\overline{\nabla^1_x f}(x_k)\|}{\sigma_k} \\ &\leq \omega_k \frac{\|\overline{\nabla^1_x f}(x_k)\|^2}{\sigma_k} \\ &= \omega_k \overline{\Delta T}^f_1(x_k,s_k) \end{aligned}$$

and (2.13) automatically holds, so that no iteration is needed in Algorithm 3.3. The resulting algorithm, where we have made the modified Step 1 explicit, is given as Algorithm 6.1 (AR1DA) on page 29.

Theorem 4.6 then guarantees that the AR1DA Algorithm will find an ϵ-approximate first-order minimizer for the unconstrained version of problem (2.1) in at most $O\big(\epsilon^{-2}\big)$ iterations and approximate evaluations of the objective function (which is proved in [13] to be optimal) and at most $O\big(|\log(\epsilon)|\epsilon^{-2}\big)$ approximate evaluations of the gradient. Note that

1. the accuracy requirement is truly adaptive and the absolute accuracy $\varepsilon_{1,i}$ may remain quite large as long as $\|\overline{\nabla_x^1 f}(x_k)\|$ itself remains large, as shown by item 3 in Step 1.

2. The accuracy requirement for computing $\overline{f}$ does not depend on the absolute accuracy of the gradient, but only on its norm (squared). At initial iterations, this may be quite large.

3. The AR1DA Algorithm is very close in spirit to the trust-region with dynamic accuracy of [17, Sections 8.4.1.1 and 10.6] and, when values of f are computed exactly, essentially recovers a proposal in [9]. It is also close to the proposal of [25], which is based on an Armijo-like linesearch and has similar accuracy requirements.

We now turn to the case where one seeks a first-order critical point for an unconstrained problem using approximate gradients and Hessians (under the assumption that the exact Hessian is Lispchitz continuous). As for the case $p = q = 1$, we have that (6.1) holds, making the verification of optimality in Step 1 relatively easy. Computing s_k is now more complicated but still practical, as it now implies minimizing the regularized quadratic model m_k starting from x_k until a step s_k is found such that

$$\|s_k\| \geq \mu\epsilon^{\frac{1}{2}} \quad \text{or} \quad \overline{\phi}^{\delta}_{m_k,1}(s_k,\omega_k) = \|\nabla_s^1 m_k(s_k)\| \leq \tfrac{1}{2}\theta\|s_k\|^2$$

(as proposed in [11], see also [21, 24, 10, 18]), with the additional constraint that, for $s_k \neq 0$,

$$\max[\varepsilon_{1,i},\varepsilon_{2,i}] \leq \omega_k \frac{\overline{\Delta T}_2^f(x_k,s_k,\omega_k)}{\chi_2(\|s_k\|)} \tag{6.7}$$

where

$$\overline{\Delta T}_2^f(x_k,s_k,\omega_k) = -\overline{\nabla_x^1 f}(x_k)^T s_k - \tfrac{1}{2}s_k^T\overline{\nabla_x^2 f}(x_k)s_k.$$

The resulting algorithm AR2DA is quite similar to AR1DA and is omitted for brevity. We note that

1. Algorithm AR2DA is guaranteed by Theorem 4.6 to find an ϵ-approximate first-order minimizer for the unconstrained version of problem (2.1) in at most $O\big(\epsilon^{-3/2}\big)$ iterations and approximate evaluations of the objective function (which is proved in [13] to be optimal) and at most $O\big(|\log(\epsilon)|\epsilon^{-3/2}\big)$ approximate evaluations of the gradient and Hessian.

2. As for AR1DA, the absolute accuracies required by AR2DA on the approximate function, gradient and Hessian only depend on the magnitude of the Taylor increment, which is typically quite large in early iterations. The relative errors on the latter two remain bounded away from zero.

3. The absolute accuracies required on the approximate gradient and Hessian are comparable in magnitude, although (6.7) could be exploited to favour one with respect to the other.

Algorithm 6.1: The AR1DA Algorithm

Step 0: Initialization. An initial point $x_0 \in \mathbb{R}^n$ and an initial regularization parameter $\sigma_0 > 0$ are given, as well as an accuracy level $\epsilon \in (0,1)$ and an initial relative accuracy $\omega_0 \geq 0$. The constants α, κ_ω, κ_ε, η_1, η_2, γ_1, γ_2, γ_3 and $\sigma_{\min}$ are also given and satisfy $\sigma_{\min} \in (0, \sigma_0]$,

$$0 < \eta_1 \leq \eta_2 < 1, \;\; 0 < \gamma_1 < 1 < \gamma_2 < \gamma_3,$$
$$\kappa_\varepsilon \in (0,1] \quad \alpha \in (0,1), \quad \kappa_\omega \in (0, \tfrac{1}{2}\alpha\eta_1] \quad \text{and} \quad \omega_0 = \min\left[\kappa_\omega, \frac{1}{\sigma_0}\right].$$

Set $k = 0$.

Step 1: Compute the optimality measure and check for termination. Initialize $\varepsilon_{1,0} = \kappa_\varepsilon$ and set $i = 0$. Do

1. compute $\overline{\nabla_x^1 f}(x_k)$ with $\|\overline{\nabla_x^1 f}(x_k) - \nabla_x^1 f(x_k)\| \leq \varepsilon_{1,i}$ and increment i by one.
2. if $\|\overline{\nabla_x^1 f}(x_k)\| \leq \epsilon/(2(1+\omega_k))$, terminate with $x_\epsilon = x_k$;
3. if $\varepsilon_{1,i} \leq \omega_k \|\overline{\nabla_x^1 f}(x_k)\|$, go to Step 2;
4. set $\varepsilon_{1,i+1} = \gamma_\varepsilon \varepsilon_{1,i}$ and return to item 1 in this enumeration.

Step 2: Step calculation. Set

$$s_k = -\overline{\nabla_x^1 f}(x_k)/\sigma_k \quad \text{and} \quad \overline{\Delta T}_1^f(x_k, s_k, \omega_k) = \|\overline{\nabla_x^1 f}(x_k)\|^2/\sigma_k.$$

Step 3: Acceptance of the trial point. Compute $\overline{f}_k(x_k + s_k, \omega_k)$ ensuring that

$$|\overline{f}_k(x_k + s_k, \omega_k) - f(x_k + s_k)| \leq \omega_k |\overline{\Delta T}_1^f(x_k, s_k, \omega_k)|. \tag{6.2}$$

Also ensure (by setting $\overline{f}_k(x_k, \omega_k) = \overline{f}_{k-1}(x_k, \omega_{k-1})$ or by (re)computing $\overline{f}_k(x_k, \omega_k)$) that

$$|\overline{f}_k(x_k, \omega_k) - f(x_k)| \leq \omega_k |\overline{\Delta T}_1^f(x_k, s_k, \omega_k)| \tag{6.3}$$

Then define

$$\rho_k = \frac{\overline{f}_k(x_k, \omega_k) - \overline{f}_k(x_k + s_k, \omega_k)}{\overline{\Delta T}_1^f(x_k, s_k, \omega_k)}. \tag{6.4}$$

If $\rho_k \geq \eta_1$, then define $x_{k+1} = x_k + s_k$; otherwise define $x_{k+1} = x_k$.

Step 4: Regularization parameter update. Set

$$\sigma_{k+1} \in \begin{cases} [\max(\sigma_{\min}, \gamma_1\sigma_k), \sigma_k] & \text{if } \rho_k \geq \eta_2, \\ [\sigma_k, \gamma_2\sigma_k] & \text{if } \rho_k \in [\eta_1, \eta_2), \\ [\gamma_2\sigma_k, \gamma_3\sigma_k] & \text{if } \rho_k < \eta_1. \end{cases} \tag{6.5}$$

Step 5: Relative accuracy update. Set

$$\omega_{k+1} = \min\left[\kappa_\omega, \frac{1}{\sigma_{k+1}}\right]. \tag{6.6}$$

Increment k by one and go to Step 1.

The case where $p = 2$ and $q = 2$ (i.e. when second-order solutions are sought) is also computationally quite accessible: calculating the optimality measure $\overline{\phi}^{\delta_k}_{f,1}(x_k, \omega_k)$ or $\overline{\phi}^{\delta_k}_{m_k,1}(s_k, \omega_k)$ now involve a standard trust-region subproblem, for which both exact and approximate numerical solvers are known (see [17, Chapter 7] for instance), but the rest of the algorithm — in particular its adaptive accuracy requirement — is very similar to what we just discussed (see also [13]). Theorem 4.6 then ensures that resulting method converges to an ϵ-approximate second-order-necessary minimizer for the unconstrained version of problem (2.1) in at most $O\big(\epsilon^{-3}\big)$ iterations and approximate evaluations of the objective function and at most $O\big(|\log(\epsilon)|\epsilon^{-3}\big)$ approximate evaluations of the gradient and Hessian.

We conclude this section by a brief discussion of the case where $q = 1$ and $p \in \{1, 2\}$ as before, but where $\mathcal{F}$ is now defined by bound constraints. It is clear that evaluating and enforcing such constraints (by projection, say) has negliglible cost and therefore falls in our framework. In this case, the calculations of $\overline{\phi}^{\delta_k}_{f,1}(x_k, \omega_k)$ or $\overline{\phi}^{\delta_k}_{m_k,1}(s_k, \omega_k)$ now involve simple linear optimization problems*, which is computationally quite tractable. If $p = 1$, Step 2.2 and 2.3 involve convex quadratic optimization, while they involve minimizing a regularized quadratic model if $p = 2$. All results remain the same, and the ARpDA algorithm is then guaranteed to find a bound-constrained approximate first-order approximate minimizer in at most $O\big(\epsilon^{-2}\big)$ or $O\big(\epsilon^{-3/2}\big)$ iterations and approximate evaluations of the objective function (which is proved in [13] to be optimal) and at most $O\big(|\log(\epsilon)|\epsilon^{-2}\big)$ or $O\big(|\log(\epsilon)|\epsilon^{-3/2}\big)$ approximate evaluations of the gradient and Hessian. The same algorithms and results obviously extend to the case where $\mathcal{F}$ is a convex polyhedral set or any closed non-empty convex set, provided the cost of the projection on this set remains negligible compared to that of (approximately) evaluating the objective function and its derivatives.

7 A stochastic viewpoint on ARpDA

7.1 Probabilistic complexity

In this section we consider the case where the bounds $\{\varepsilon_j\}_{j=1}^p$ on the absolute errors on the derivative tensors $\{\nabla_x^j f(x)\}_{j=1}^p$ are satisfied with probability at least $(1 - t)$, with $t \in (0, 1)$. This may occur, for instance, if the approximate derivative tensors are obtained by some stochastic sampling scheme, as we detail below. We therefore assume that

$$Pr\Big[\|\overline{\nabla_x^j f}(x_k) - \nabla_x^j f(x_k)\|_{[j]} \leq \varepsilon_j\Big] \geq (1 - t) \quad \text{for each} \quad j \in \{1, \ldots, p\}. \tag{7.1}$$

We also assume that inequalities (2.21) and (2.22) in Step 3 of the ARpDA algorithm are satisfied with probability at least $(1 - t)$, i.e.

$$Pr\Big[|\overline{f}_k(x_k + s_k, \omega_k) - f(x_k + s_k)| \leq \varepsilon_0\Big] \geq 1 - t, \tag{7.2}$$

and

$$Pr\Big[|\overline{f}_k(x_k, \omega_k) - f(x_k)| \leq \varepsilon_0\Big] \geq 1 - t \tag{7.3}$$

where we have defined $\varepsilon_0 \stackrel{\rm def}{=} \omega_k|\overline{\Delta T}_p^f(x_k, s_k, \omega_k)|$. Clearly, different values for t could be chosen in (7.1), one for each index (tensor order) $j \in \{1, ..., p\}$. Similarly, different values of

*Formerly known as linear programming problems, or LPs.

t in (7.2) and (7.3) could be considered. However, for the sake of simplicity, we assume here that all the inequalities involved in (7.1)–(7.3) hold with the same fixed lower bound $(1-t)$ on the probability of success. We also assume that the events in (7.1)–(7.3) are independent.

Stochastic variants of trust-region and adaptive cubic regularization methods have been analyzed in [2, 8, 15, 30, 32]. In [8, 15], complexity results are given in expectation, while the analysis is carried out in probability in [2, 30, 32]. We choose to follow the high-probability approach of [30, 32], where an overall and cumulative success of (7.1)–(7.3) is assumed along all the iterations up to termination.

We stress that Algorithms 3.2 and 3.3 terminates independently of the satisfaction of the accuracy requirements on the tensor derivatives. This is due to the fact that termination relies on the inequality (3.6). Moreover, during the iterations of either of these algorithms before the last, it may happen that the accuracy on the tensor derivatives fails to be achieved, but this has no impact on the worst-case complexity. Satisfying the accuracy requirement is only crucial in the last iteration of Algorithm 3.2 or 3.3 (that is in Steps 1.2 and 2.2). Let $\mathcal{E}_r(S)$ be the event: "the relations

$$\|\overline{\nabla_x^j f}(x_k) - \nabla_x^j f(x_k)\|_{[j]} \leq \varepsilon_j \;\; \text{for all} \;\; j \in \{1,\ldots,r\}$$

hold for some j at Step S of the last iteration of the relevant algorithm". In Step 1.2, inexact values are computed for the first q derivatives, and the probability that event $\mathcal{E}_q(1.2)$ occurs is therefore at least $(1-t)^q$. Similarly, the probability that event $\mathcal{E}_q(2.2)$ occurs is at least $(1-t)^p$. Finally, at Step 3 of the ARpDA algorithm, the probability that both (2.21) and (2.22) hold is at least $(1-t)^2$. Then, letting for $i \in \{1,\ldots,k\}$, $\mathcal{E}_{[i]}$ be the event: "Inequalities (2.13), (2.21) and (2.22) hold at iteration i, of the ARpDA algorithm", the probability that $\mathcal{E}_{[i]}$ occurs is then at least $(1-t)^{p+q+2}$. Finally, letting $\mathcal{E}(k)$ be the event: "$\mathcal{E}_{[i]}$ occurs for all iterations $i \in \{1,\ldots,k\}$ of the ARpDA algorithm", we deduce that

$$Pr\Big[\mathcal{E}(k)\Big] \equiv Pr\left[\bigcap_{i=1}^{k} \mathcal{E}_{[i]}\right] \geq (1-t)^{k(p+q+2)}.$$

Thus, requiring that the event $\mathcal{E}(k)$ occurs with probability at least $1-\overline{t}$, we obtain that

$$Pr\Big[\mathcal{E}(k)\Big] \geq (1-t)^{k(p+q+2)} = 1-\overline{t}, \;\; \text{i.e.,} \;\; t = 1-(1-\overline{t})^{\frac{1}{k(p+q+2)}} = O\left(\frac{\overline{t}}{k(p+q+2)}\right).$$

Taking into account that, when (2.13), (2.21) and (2.22) hold, the ARpDA algorithm terminates in at most $k = O\big(\epsilon^{-\frac{p+\beta}{p-q+\beta}}\big)$ iterations (as stated by Theorem 4.6), we deduce the following result.

Theorem 7.1 Let $f \in \mathcal{C}^{p,\beta}(\mathbb{R}^n)$. Suppose that the probabilistic assumptions of this section hold and that, at each of iteration of the ARpDA algorithm, the probability t satisfies

$$t = O\left(\frac{\overline{t}\,\epsilon^{\frac{p+\beta}{p-q+\beta}}}{(p+q+2)}\right). \tag{7.4}$$

Then, given $\epsilon \in (0,1)$, the conclusions of Theorem 4.6 hold with probability at least $(1-\overline{t})$.

As a consequence, when $p = q = 2$ and $\beta = 1$ we have to choose $t = O\left(\frac{1}{6}\,\overline{t}\,\epsilon^3\right)$, while, when $p = q = \beta = 1$, we have to choose $t = O\left(\frac{1}{4}\,\overline{t}\,\epsilon^2\right)$.

We stress that the above analysis is unduly pessimistic in the case where $p = q = 1$. Indeed, as already noticed in Section 6, no reduction in $\{\varepsilon_j\}$ is necessary at Step 2, as (2.13) is automatically enforced whenever the relative error on the first derivative $\overline{\nabla_x^1 f}(x)$ is bounded by ω_k. Noting that this last event has probability at least $1 - t$, we can conclude that $Pr(\mathcal{E}_{[i]}) \geq (1 - t)^3$ and to get the optimal complexity $O(\epsilon^{-2})$ with probability at least $1 - \overline{t}$, we need to choose $t = O\left(\frac{1}{3}\,\overline{t}\,\epsilon^2\right)$. We also emphasize that the purpose of Theorem 7.1 is limited to offer guidance on desirable value of t and not to prescribe an algorithmically binding bound. Indeed some of the constants involved in the bound of Theorem 4.6 (and thus of Theorem 7.1) are typically unknown a priori (which is why we have not been more specific in (7.4)).

7.2 Sample size in subsampling for finite-sum problems

In what follows, we now focus on the solution of large-scale instances of the finite-sum problems arising in machine learning and data analysis, that are modelled as

$$\min_{x\in\mathcal{F}} f(x) = \frac{1}{N}\sum_{i=1}^{N} \psi_i(x), \tag{7.5}$$

with $N > 0$ and $\psi_i : \mathbb{R}^n \to \mathbb{R}$. Restricting ourselves to the cases where $p \leq 2$, we discuss the application of Algorithm AR1DA and AR2DA to problem (7.5). In this case, the approximation of the objective function's value and of first and second derivatives is obtained by a subsampling procedures, i.e. these quantities are approximated by randomly sampling component functions ψ_i. More precisely, at iteration k these approximations take the form:

$$\overline{f}_k(x_k, \omega_k) = \frac{1}{|\mathcal{D}_{k,1}|}\sum_{i\in\mathcal{D}_{k,1}} \psi_i(x_k), \quad \overline{f}_k(x_k + s_k, \omega_k) = \frac{1}{|\mathcal{D}_{k,2}|}\sum_{i\in\mathcal{D}_{k,2}} \psi_i(x_k + s_k),$$

$$\overline{\nabla_x^1 f}(x_k) = \frac{1}{|\mathcal{G}_k|}\sum_{i\in\mathcal{G}_k} \overline{\nabla_x^1 \psi_i}(x_k), \quad \text{and} \quad \overline{\nabla_x^2 f}(x_k) = \frac{1}{|\mathcal{H}_k|}\sum_{i\in\mathcal{H}_k} \overline{\nabla_x^2 \psi_i}(x_k),$$

where $\mathcal{D}_{k,1}$, $\mathcal{D}_{k,2}$, $\mathcal{G}_k$ and $\mathcal{H}_k$ are subsets of $\{1, 2, \ldots, N\}$. The question then arises of estimating the cardinality of these sample sets in order to ensure that the approximations of the objective function's value and its first and second derivatives satisfy (7.1) for $j = 1$ and $j = 2$, (7.2) and (7.3). This issue can be addressed using the operator-Bernstein inequality given in [28] and recently extended in [3] to random tensors of general order. In the next theorem we derive our final result concerning the sample sizes for subsampling the objective function and its derivatives up to order two.

Theorem 7.2 Suppose that there exist non-negative constants $\{\kappa_{\psi,j}\}_{j=0}^{2}$ such that, for $x \in \mathbb{R}^n$ and all $j \in \{0,1,2\}$

$$\max_{i\in\{1,\ldots,N\}} \|\nabla_x^j \psi_i(x)\| \leq \kappa_{\psi,j}(x). \tag{7.6}$$

Let $t \in (0,1)$ and suppose that a subsample $\mathcal{A}_k$ is chosen randomly and uniformly from $\{1,\ldots,N\}$ and that, for some $j \in \{0,1,2\}$, one computes

$$\overline{\nabla_x^j f}(x) = \frac{1}{|\mathcal{A}_k|} \sum_{i\in\mathcal{A}_k} \overline{\nabla_x^j \psi_i}(x),$$

with

$$|\mathcal{A}_k| \geq \min\left\{N, \left\lceil \frac{4\kappa_{\psi,j}(x)}{\varepsilon_j}\left(\frac{2\kappa_{\psi,j}(x)}{\varepsilon_j}+\frac{1}{3}\right)\log\left(\frac{d}{t}\right)\right\rceil\right\}, \tag{7.7}$$

where

$$d = \begin{cases} 2, & \text{if } j=0,\\ n+1, & \text{if } j=1,\\ 2n, & \text{if } j=2. \end{cases}$$

Then condition (7.1) holds for $x = x_k$ with probability at least $(1-t)$ if $j \in \{1,2\}$, or, if $j = 0$, each of the conditions (7.2) and (7.3) holds with probability at least $(1-t)$ for $x = x_k + s_k$ and $x = x_k$, respectively.

Proof. L □

et $\mathcal{A}_k \subseteq \{1,\ldots,N\}$ be a sample set of cardinality $|\mathcal{A}_k|$. Consider $j \in \{0,1,2\}$ and $|\mathcal{A}_k|$ random tensors $Z_u(x)$ such that,

$$Pr\left[Z_u(x) = \nabla_x^j \psi_i(x)\right] = \frac{1}{N}, \quad (i \in \{1,\ldots,N\}).$$

For $u \in \mathcal{A}_k$, let us define

$$X_u \stackrel{\text{def}}{=} \left(Z_u(x) - \nabla_x^j f(x)\right), \qquad \overline{\nabla_x^j f}(x) \stackrel{\text{def}}{=} \frac{1}{|\mathcal{A}_k|}\sum_{u\in\mathcal{A}_k} Z_u(x)$$

and

$$X \stackrel{\text{def}}{=} \sum_{u\in\mathcal{A}_k} X_u = |\mathcal{A}_k|\left(\overline{\nabla_x^j f}(x) - \nabla_x^j f(x)\right).$$

Since (7.5) gives that

$$\frac{1}{N}\sum_{i=1}^{N} \nabla_x^j \psi_i(x) = \nabla_x^j f(x),$$

we deduce that

$$E(X_u) = \frac{1}{N}\sum_{i=1}^{N}\left(\nabla_x^j \psi_i(x) - \nabla_x^j f(x)\right) = 0, \quad u \in \mathcal{A}_k.$$

Moreover, assuming $Z_u(x) = \nabla_x^j \psi_l(x)$ for some $l \in \{1, \ldots, N\}$ and using (7.6), we have that

$$\|X_u\| \leq \left\| \frac{N-1}{N} \nabla_x^j \psi_l(x) - \frac{1}{N} \sum_{i \in \{1,\ldots,N\} \setminus \{l\}} \nabla_x^j \psi_i(x) \right\| \leq 2 \frac{N-1}{N} \kappa_{\psi,j}(x) \leq 2\kappa_{\psi,j}(x),$$

so that the variance of X can be bounded as follows:

$$\begin{aligned} v(X) &= \max\left[\|E(XX^T)\|, \|E(X^TX)\|\right] \\ &= \max\left[\Big\| \sum_{u \in \mathcal{A}_k} E(X_u X_u^T)\Big\|, \Big\| \sum_{u \in \mathcal{A}_k} E(X_u^T X_u)\Big\|\right] \\ &\leq \max\Big[\sum_{u \in \mathcal{A}_k} \|E(X_u X_u^T)\|, \sum_{u \in \mathcal{A}_k} \|E(X_u^T X_u)\|\Big] \\ &\leq \max\Big[\sum_{u \in \mathcal{A}_k} E(\|X_u X_u^T\|), \sum_{u \in \mathcal{A}_k} E(\|X_u^T X_u\|)\Big] \\ &\leq \sum_{u \in \mathcal{A}_k} E(\|X_u\|^2) \leq 4|\mathcal{A}_k|\kappa_{\psi,j}^2(x), \end{aligned}$$

in which the first and the third inequalities hold because of the triangular inequality, while the second is due to the Jensen's inequality (note that the spectral norm $\|\cdot\|$ is convex). Therefore, according to the Operator-Bernstein Inequality stated in [28, Theorem 6.1.1], we obtain that

$$Pr\Big[\|\overline{\nabla_x^j f}(x) - \nabla_x^j f(x)\| \geq \epsilon_j\Big] = Pr\Big[\|X\| \geq \epsilon_j |\mathcal{A}_k|\Big] \leq d\, e^{-\frac{\epsilon_j^2 |\mathcal{A}_k|}{4\kappa_{\psi,j}(x)\left(2\kappa_{\psi,j}(x) + \frac{1}{3}\epsilon_j\right)}}, \tag{7.8}$$

with $d = 2$ if $j = 0$, $d = n+1$ if $j = 1$ and $d = 2n$ if $j = 2$. Then, bounding the right-hand side of (7.8) by t, taking logarithms and extracting $|\mathcal{A}_k|$ gives (7.7).

In particular, Theorem 7.2 gives the lower bounds

$$|\mathcal{D}_{k,\ell}| \geq \min\left\{N, \left\lceil \frac{4\kappa_{\psi,j}(x)}{\varepsilon_0}\left(\frac{2\kappa_{\psi,j}(x)}{\varepsilon_0} + \frac{1}{3}\right)\log\left(\frac{2}{t}\right)\right\rceil\right\}, \quad \ell = 1, 2, \tag{7.9}$$

$$|\mathcal{G}_k| \geq \min\left\{N, \left\lceil \frac{4\kappa_{\psi,j}(x)}{\varepsilon_1}\left(\frac{2\kappa_{\psi,j}(x)}{\varepsilon_1} + \frac{1}{3}\right)\log\left(\frac{n+1}{t}\right)\right\rceil\right\} \tag{7.10}$$

and

$$|\mathcal{H}_k| \geq \min\left\{N, \left\lceil \frac{4\kappa_{\psi,j}(x)}{\varepsilon_2}\left(\frac{2\kappa_{\psi,j}(x)}{\varepsilon_2} + \frac{1}{3}\right)\log\left(\frac{2n}{t}\right)\right\rceil\right\}. \tag{7.11}$$

The adaptive nature of these sample sizes is apparent in formulae (7.9)–(7.11), because they depend on x and ε_j, which are themselves dynamically updated in the course of the ARpDA algorithm. Depending on the size of N, it may clearly be necessary to consider the whole set $\{1, \ldots, N\}$ for small values of $\{\varepsilon_j\}_{j=0}^2$. If the cost of evaluating functions ψ_i, for $1 \leq i \leq N$, is comparable for all i, the cost of evaluating $\overline{f}_k(x_k, \omega_k)$ amounts to the fraction $|\mathcal{D}_{k,1}|/N$ of the effort for computing the exact value $f(x_k)$. Analogous considerations hold for the objective function's derivatives.

The implementation of rules (7.9)-(7.11) requires the knowledge of the size of the functions ψ_i's and their first and second order derivatives. If only global information is available, the

dependence on x may obviously be avoided by choosing a uniform upper bound $\kappa_{\psi,j}$ for all $x \in \mathcal{F}$, at the cost of a lesser adaptivity. Similar bounds on the sample size used to approximate gradients and Hessians up to a prescribed probability have been derived and used in [26] where it has also been observed that there are problems where estimations of the needed uniform upper bounds can be obtained. In particular, let $\{(a_i, b_i)\}_{i=1}^N$ denote the pairs forming a data set with $a_i \in \mathbb{R}^n$ being the vector containing the features of the i-th example and b_i being its label. In [26] authors considered the minimization of objective function $(1/N)\sum_{i=1}^N(\Phi(a_i^T x) - b_i a_i^T x)$ over a sparsity inducing constraint set, e.g., $\mathcal{F} = \{x \in \mathbb{R}^n \mid \|x\|_1 \leq 1\}$, for cumulant generating functions Φ of different forms, and explicitly provided the uniform bound $\kappa_{\psi,1}$. Taking into account that x belongs to the set $\mathcal{F}$, uniform bounds for the objective function and the Hessian norm can also be derived.

Uniform bounds are available also in the unconstrained setting for binary classification problems modelled by the sigmoid function and least-squares loss, i.e. problems of the form (7.5) with $\mathcal{F} \equiv \mathbb{R}^n$ and

$$\psi_i(x) = \left(b_i - \frac{1}{1 + e^{-a_i^T x}}\right)^2, \quad i = 1 \ldots, N. \tag{7.12}$$

Let $v_i(x) = (1 + e^{-a_i^T x})^{-1}$ and note that $b_i \in \{0, 1\}$ and $v_i(x) \in (0, 1)$ for any $x \in \mathbb{R}^n$. Then, $|\psi_i(x)| \leq 1$, for any $x \in \mathbb{R}^n$. Moreover, uniform upper bounds $\kappa_{\psi,j}$ for $\nabla_x^j \psi_i(x)$, $j = 1, 2$ can be easily derived and they are reported Table 1 along with the expression of the first and second order derivatives of $\psi_i(x)$. The computation of these bounds requires a pre-processing phase as the norm of the features vectors $\{a_i\}_{i=1}^N$ of the data sets are needed.

	Derivatives	$\kappa_{\psi,j}$
$\nabla_x \psi_i(x)$	$-(b_i - v_i(x))(1 - v_i(x))v_i(x)a_i$	$\frac{1}{5}\|a_i\|$
$\nabla_x^2 \psi_i(x)$	$v_i(x)(1 - v_i(x))(3v_i(x)^2 - 2v_i(x)(1 + b_i) + b_i)a_i a_i^T$	$\frac{1}{10}\|a_i\|^2$

Table 1: First and second order derivatives of (7.12) and corresponding uniform bounds

Finally, whenever N is large enough to ensure that (7.9)–(7.11) do not require the full sample, the size of the sample used to obtain a single approximate objective function value is $O(\varepsilon_0^{-2})$. Analogously, gradient and Hessian values are approximated by averaging over samples of size $O(\varepsilon_1^{-2})$ and $O(\varepsilon_2^{-2})$, respectively. In Step 3 of the AR1DA algorithm, the choice $\varepsilon_0 \in \left[\gamma_\epsilon \omega_k \|\overline{\nabla_x^j f}(x)\|^2/\sigma_k, \omega_k \|\overline{\nabla_x^j f}(x)\|^2/\sigma_k\right]$ is required to ensure that (6.2)-(6.3) are satisfied. With this choice, iteration k of the AR1DA algorithm requires $O(\|\overline{\nabla_x^j f}(x)\|^{-4})$ ψ_i-evaluations ($O(\epsilon^{-4})$ ψ_i-evaluations in the worst case). Similarly, $\varepsilon_0 = O(\omega_k \|\overline{\Delta T}_2^f(x_k, s_k, \omega_k)\|)$ is needed at iteration k of the AR2DA algorithm. As a consequence, and if the algorithm does not terminate at iteration $k + 1$, it follows from Lemma 4.1 and 4.4 that $O(\epsilon^{-(3/(3-q))^2})$ ψ_i-evaluations may be required in the worst case. Finally, using Lemma 3.5 and (3.29), we claim that each iteration of the AR1DA and AR2DA algorithms requires at most $O((1+\nu_{\max}(\epsilon))\epsilon^{-2})$ evaluations of component gradients and component Hessians, where $\nu_{\max}(\epsilon)$ has been defined in (3.30). These bounds turn out to be better or the same as those derived in [8],[15],[32]. Although they may appear discouraging, it should be kept in mind that they are valid only if N is truly large compared with $1/\epsilon$ (for instance, it has to exceed $O(\epsilon^{-4})$ to allow for approximate functions in the AR1DA Algorithm). In other words, the sampling schemes (7.9)–(7.11) are most relevant when $1/\epsilon$ remains modest compared with N.

We conclude by emphasizing that the per-iteration failure probability t given in (7.4) is not too demanding in what concerns the sample size, because it only occurs in the logarithm term of (7.7). The same is true of the impact of the value of the unknown constants hidden in the $O(\cdot)$ notation in (7.4).

8 Conclusion and perspectives

We have provided a general regularization algorithm using inexact function and derivatives' values, featuring a flexible adaptive mechanism for specifying the amount of inexactness acceptable at each iteration. This algorithm, inspired by the unifying framework proposed in [13], is applicable to unconstrained and inexpensively-constrained nonconvex optimization problems, and provides optimal iteration complexity for arbitrary degree of available derivatives, arbitrary order of optimality and the full range of smoothness assumptions on the objective function highest derivative. We have also specialized this algorithm to the cases of first- and second-order methods, exhibiting simple and numerically realistic methods. We have finally provided a probabilistic version of the complexity analysis and derived associated lower bounds on sample size in the context of subsampling methods.

There are of course many ways in which the proposed algorithm might be improved. For instance, the central calculation of relatively accurate Taylor increments may possibly be made more efficient by updating the absolute accuracies for different degrees separately. Further techniques to avoid unnecessary derivative computations (without affecting the optimal complexity) could also be investigated.

The framework proposed in this paper also offers obvious avenues for specializations to specific contexts, among which we outline two. The first is that of algorithms using stochastic approximations of function values and derivatives. The technique presented here derives probabilistic conditions under which properties of the deterministic algorithms are preserved. It does not provide an algorithm which is robust against failures to satisfy the adaptive accuracy requirements. This is in contrast with the interesting analysis of unconstrained first-order methods of [25] and [8]. Combining the generality of our approach with the robustness of the proposal in these latter papers is thus desirable. The second interesting avenue is the application of the new results to multi-precision optimization in the context of very high performance computing. In this context, it is of paramount importance to limit energy dissipation in the course of an accurate calculation, and this may be obtained by varying the accuracy of the most crucially expensive of its parts (see [20] for unconstrained quadratic optimization). The discussion above again provides guidance at what level of arithmetic accuracy is needed to achieve overall performance while maintaining optimal complexity. Both these topics are the object of ongoing research and will be reported on at a later stage.

Acknowledgments

INdAM-GNCS partially supported the first and third authors under Progetti di Ricerca 2018. The second author was partially supported by INdAM through a GNCS grant. The last author gratefully acknowledges the support and friendly environment provided by the Department of Industrial Engineering at the Università degli Studi, Florence (Italy) during his visit in the fall of 2018. The authors are also indebted to two careful referees, whose comments and perceptive questions have resulted in a significant improvement of the manuscript.

References

[1] L. Bottou, F. E. Curtis and J. Nocedal. Optimization Methods for Large-Scale Machine Learning. *SIAM Review*, 60:233-311, 2018.

[2] A. Bandeira, K. Scheinberg and L. Vicente. Convergence of trust-region methods based on probabilistic models. *SIAM Journal on Optimization*, 24:1238:1264, 2014.

[3] Z. Luo, L. Qi and Ph. L Toint. Bernstein Concentration Inequalities for Tensors via Einstein Products. arXiv:1902.03056, 2018.

[4] S. Bellavia, S. Gratton, and E. Riccietti. A Levenberg-Marquardt method for large nonlinear least-squares problems with dynamic accuracy in functions and gradients. *Numerische Mathematik*, 140:791–825, 2018.

[5] S. Bellavia, G. Gurioli, and B. Morini. Theoretical study of an adaptive cubic regularization method with dynamic inexact Hessian information. arXiv:1808.06239, 2018.

[6] E. Bergou, Y. Diouane, V. Kungurtsev, and C. W. Royer. A subsampling line-search method with second-order results. arXiv:1810.07211, 2018.

[7] 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, Series A*, 163(1):359–368, 2017.

[8] J. Blanchet, C. Cartis, M. Menickelly, and K. Scheinberg. Convergence rate analysis of a stochastic trust region method via supermartingales. arXiv:1609.07428v3, 2018.

[9] T. Bonniot. Convergence and complexity of unconstrained optimization methods with inexact gradients. Master's thesis, ENSEEIHT, Toulouse, France, September 2018. (supervised by S. Gratton, D. Orban and Ph. Toint).

[10] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Trust-region and other regularization of linear least-squares problems. *BIT*, 49(1):21–53, 2009.

[11] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Adaptive cubic overestimation methods for unconstrained optimization. Part II: worst-case function-evaluation complexity. *Mathematical Programming, Series A*, 130(2):295–319, 2011.

[12] C. Cartis, N. I. M. Gould, and Ph. L. Toint. On the oracle complexity of first-order and derivative-free algorithms for smooth nonconvex minimization. *SIAM Journal on Optimization*, 22(1):66–86, 2012.

[13] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Sharp worst-case evaluation complexity bounds for arbitrary-order nonconvex optimization with inexpensive constraints. arXiv:1811.01220, 2018.

[14] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Worst-case evaluation complexity and optimality of second-order methods for nonconvex smooth optimization. To appear in the Proceedings of the 2018 International Conference of Mathematicians (ICM 2018), Rio de Janeiro, 2018.

[15] C. Cartis and K. Scheinberg. Global convergence rate analysis of unconstrained optimization methods based on probabilistic models. *Mathematical Programming, Series A*, 159(2):337–375, 2018.

[16] X. Chen, B. Jiang, T. Lin, and S. Zhang. On adaptive cubic regularization Newton's methods for convex optimization via random sampling. arXiv:1802.05426, 2018.

[17] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, USA, 2000.

[18] J. P. Dussault. Simple unified convergence proofs for the trust-region and a new ARC variant. Technical report, University of Sherbrooke, Sherbrooke, Canada, 2015.

[19] S. Gratton, A. Sartenaer, and Ph. L. Toint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization*, 19(1):414–444, 2008.

[20] S. Gratton, E. Simon, and Ph. L. Toint. Minimizing convex quadratics with variable precision Krylov methods. arXiv:1807.07476, 2018.

[21] 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.

[22] L. Liu, X. Liu, C.-J. Hsieh, and D. Tao. Stochastic second-order methods for non-convex optimization with inexact Hessian and gradient. arXiv:1809.09853, 2018.

[23] Yu. Nesterov. *Introductory Lectures on Convex Optimization*. Applied Optimization. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2004.

[24] Yu. Nesterov and B. T. Polyak. Cubic regularization of Newton method and its global performance. *Mathematical Programming, Series A*, 108(1):177–205, 2006.

[25] C. Paquette and K. Scheinberg. A stochastic line search method with convergence rate analysis. arXiv:1807.07994, 2018.

[26] F. Roosta-Khorasani, M. W. Mahoney. Sub-sampled Newton methods. *Mathematical Programming*, 174(1-2):293–326, 2019.

[27] N. Tripuraneni, M. Stern, J. Regier, and M. I. Jordan. Stochastic cubic regularization for fast nonconvex optimization. arXiv:1711.02838v2, 2017.

[28] J. Tropp. *An Introduction to Matrix Concentration Inequalities*. Number 8:1-2 in Foundations and Trends in Machine Learning. Now Publishing, Boston, USA, 2015.

[29] S. A. Vavasis. Black-box complexity of local minimization. *SIAM Journal on Optimization*, 3(1):60–80, 1993.

[30] P. Xu, F. Roosta-Khorasani, and M. W. Mahoney. Newton-type methods for non-convex optimization under inexact Hessian information. arXiv:1708.07164v3, 2017.

[31] P. Xu, F. Roosta-Khorasani, and M. W. Mahoney. Second-order optimization for non-convex machine learning: An empirical study. arXiv:1708.07827v2, 2018.

[32] Z. Yao, P. Xu, F. Roosta-Khorasani, and M. W. Mahoney. Inexact non-convex Newton-type methods. arXiv:1802.06925, 2018.