Inexact restoration with subsampled trust-region methods for finite-sum minimization[‡]

Stefania Bellavia[*], Nataša Krejić[†], Benedetta Morini[*]

May 10, 2020

Abstract

Convex and nonconvex finite-sum minimization arises in many scientific computing and machine learning applications. Recently, first-order and second-order methods where objective functions, gradients and Hessians are approximated by randomly sampling components of the sum have received great attention.

We propose a new trust-region method which employs suitable approximations of the objective function, gradient and Hessian built via random subsampling techniques. The choice of the sample size is deterministic and ruled by the inexact restoration approach. We discuss local and global properties for finding approximate first- and second-order optimal points and function evaluation complexity results. Numerical experience shows that the new procedure is more efficient, in terms of overall computational cost, than the standard trust-region scheme with subsampled Hessians.

Keywords: inexact restoration, trust-region methods, subsampling, local and global convergence, worst-case evaluation complexity.

1 Introduction

The problem we consider in this paper is the following

$$\min_{x \in \mathbb{R}^n} f_N(x) = \frac{1}{N} \sum_{i=1}^{N} \phi_i(x), \tag{1}$$

[*] Dipartimento di Ingegneria Industriale, Università degli Studi di Firenze, Viale G.B. Morgagni 40, 50134 Firenze, Italia. Members of the INdAM Research Group GNCS. Emails: stefania.bellavia@unifi.it, benedetta.morini@unifi.it

[†] Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Trg Dositeja Obradovića 4, 21000 Novi Sad, Serbia, Email: natasak@uns.ac.rs.

[‡] The work of Bellavia and Morini was supported by Gruppo Nazionale per il Calcolo Scientifico (GNCS-INdAM) of Italy. The work of the second author was supported by Serbian Ministry of Education, Science and Technological Development, grant no. 174030.

Part of the research was conducted during a visit of the second author at Dipartimento di Ingegneria Industriale supported by Piano di Internazionalizzazione, Università degli Studi di Firenze.

where N is very large and finite and $\phi_i : \mathbb{R}^n \to \mathbb{R}$. A number of important problems can be stated in this form, to start with problems in machine learning like classification problems, data fitting problems, sample average approximation of the objective function given in the form of mathematical expectation and so on.

The practical relevance of (1) resulted in a number of methods that are adjusted to this particular form of the objective function. In fact, for very large N the cost of evaluating f_N might be really high and the same is true for the gradient and even more for the Hessian evaluation. Therefore a number of methods that use approximate objective functions and/or first and second order derivatives, formed by partial sums, is proposed and analysed in literature, see e.g., [3–6, 8–13, 19, 20, 25, 36–38].

Concerning the approximation of the objective function, one of the possible approaches is to use relatively rough approximations at early stages of the optimization procedure and gradually increase the accuracy to arrive at full precision at the late stage of the iterative procedure; the gradient is approximated accordingly. This way one hopes to save computational effort and yet to solve the original problem eventually. Very often the term scheduling is used to describe the approximation of the objective function by means of a partial sum. There is a number of algorithms proposed for the scheduling problem, ranging from simple heuristics that increase the number of terms in the partial sum that approximates the objective function by a certain percentage in each iteration, [5, 10, 20, 36] to more elaborate schemes that connect the progress achieved during the optimization procedure to the number of terms in the partial sum [1–5, 7–9, 13, 17, 27–29, 33, 35].

Besides the problem of scheduling, one has to decide between first- and second-order optimization method to be employed. A detailed survey is presented in [11]. A number of first-order methods has been proposed and analysed in the literature. Given that the main cost comes from large N one might be tempted to conclude that computing Hessians, or some other second order information might be prohibitively costly and thus opt for a first order method, especially if the problem (1) should be solved with limited precision. However, recently there has been reported in several papers that careful adjustment and implementation of second order methods might be worth considering if the true Hessian is approximated by a partial sum of Hessians $\nabla^2 \phi_i(x)$ consisting of a significantly smaller number of terms than N. This way one can generate useful information with significantly smaller cost than the true Hessian and get enough advantage over first-order methods in terms of resilience to problem ill-conditioning and low sensitivity to parameter tuning, [5, 6, 10, 12, 13, 19, 34, 36–38].

The method we present here combines the Inexact Restoration (IR) framework with the trust-region optimization method [16] to simultaneously design the scheduling and the optimization procedure for solving (1) and represents a new approach for the problem under consideration.

The Inexact Restoration method, introduced in [31], is a constrained optimization tool particularly suitable for problems where one does not want to enforce feasibility in all iterations. The key idea of the IR approach is to treat feasibility and optimality in a modular way and to improve each one in separate procedures; the combination of feasibility and optimality is then monitored through a suitable merit function. Each iteration ensures the sufficient decrease of a suitable merit function and therefore, under certain assumption, convergence to a feasible optimal point. In [30, 31] the combination of the IR strategy with trust-region methods is proposed and analysed for general constrained problems.

The application of IR strategy to the unconstrained optimization problem (1) requires its reformulation as a constrained problem. Letting I_M be an arbitrary nonempty subset of $\{1, \ldots, N\}$ of cardinality $|I_M|$ equal to M, we reformulate problem (1) as

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} f_M(x) = \frac{1}{M} \sum_{i \in I_M} \phi_i(x). \\ &\text{s.t. } M = N, \end{aligned} \tag{2}$$

Evaluating infeasibility in (2) is cheap while computing the objective function is expensive whenever M is large. Thus, using the reasoning from [30,31] we define a new algorithm that exploits the structure of the problem considered and takes advantage of the modular structure of IR and the trust-region optimization method at the same time. Specifically, the trust-region mechanism is applied to model f_M at each iteration and the IR framework is applied to test for the acceptance of the iterates and to determine the scheduling sequence, i.e. the value of M through the iterations. The test acceptance of the new iterate allows us to deal with inaccuracy in function and derivatives. In particular, the number of terms in the partial sum is fixed at the beginning of each iteration in the restoration phase and possibly changed in the optimality phase where the trial iterate is computed.

Clearly, the higher feasibility is the more accurate f_M is with respect to f_N. The new procedure has two important properties: partial sums, possibly consisting of small sets of ϕ_i's, can be used in the early stage of the iterative procedure to decrease the computational cost; the original objective function in (1) is recovered for all iteration indices large enough, thus allowing for the solution of the given problem. Clearly, when full precision of the objective function and the gradient is reached, one can rely on the theory and machinery of standard trust-region methods [16].

The scheme presented here applies to both first- and second-order trust-region models. If a linear model is used, the resulting procedure is a subsampled gradient method with variable stepsize. When second-order models are used, the Hessian can be approximated using a subset of the sample used to approximate function and gradient. The error in such Hessian approximation plays an important role in the asymptotic convergence rate. In the case of strongly convex problems, the analysis for local linear convergence rate is presented, both in deterministic and probabilistic settings, and an adaptive choice of the sample for Hessian approximation is proposed.

We also provide a function evaluation complexity result which resembles the classical result for the trust-region methods for (1) and the results obtained in [8]. It is shown that at most $O(\varepsilon^{-2})$ evaluations of the possibly subsampled function f_M, $M \leq N$, and its derivatives are needed to compute a first-order approximate critical point. Then the worst-case complexity of the standard trust-region is recovered with expected significant computational savings due to scheduling.

Our approach considerably differs from the IR procedure and trust-region method in [30, 31] since the objective function in our formulation changes with M through the iterations. It also differs from IR approaches in [7, 8, 29] that employ approximate objective function and its derivatives and have been successfully applied to constrained and unconstrained problems, including problem (1); in papers [7, 29] the IR is combined with a line search strategy, while in [8] the considered problem is constrained and regularization techniques are used in the optimization phase. The approach presented here relays on [8] in terms of general idea but the problem is more specific being a finite-sum rather than a general objective function computed approximately and being unconstrained. These specifications allow us to design an efficient sample update rule which is connected with the trust-region size.

The value of M is fixed via a deterministic rule while the trust-region schemes in [9,25,38], approximating either functions, gradients and Hessians [9,25] or Hessians only [38], are designed using sample sets whose cardinality is determined by high probability and nonasymptotic convergence analysis.

The nature of IR allows changes in the feasibility through iterations and the change is not necessarily monotone, i.e., the cardinality of the subset that defines the approximate objective can both increase and decrease, depending on the feedback from the trust-region progress made in each iteration. The case where M is increased by a prefixed percentage at each iteration is a particular case of our strategy. In this latter case our method differs from a straightforward subsampled trust-region procedure with increasing sample size in both the merit function and the acceptance criterion. Remarkably, their employment allow to prove optimal complexity results that otherwise require adaptive accuracy requirements [9].

This paper is organized as follows. In Section 2 we present our method and prove that it is well defined. Furthermore, we prove that full accuracy is eventually reached and that the set of standard assumptions yield first-order stationary points. Some issues concerning the realization of the procedure are considered in Section 3; the scheduling rule is modified to avoid unproductive decrease in precision and a discussion on first and second order trust-region models is provided. Section 4 deals with strongly convex problems; we prove q-linear convergence as well as q-linear convergence in expectation under probabilistic bounds for Hessian subsampling. Section 5 provides worst-case function evaluation complexity. The numerical performance of the proposed method is tested on a set of classification problems and the results are reported in Section 6.

2 The Algorithm

Let I_M be an arbitrary nonempty subset of $\{1,\dots,N\}$ of cardinality $|I_M|$ equal to M,

$$I_M \subseteq \{1,\dots,N\}, \quad |I_M| = M, \quad M \geq 1,$$

and reformulate (1) as the constrained problem (2). We measure the level of infeasibility with respect to the constraint $M = N$ by the function h with the following properties.

Assumption 2.1 *Let $h : \mathbb{N} \to \mathbb{R}$ be a monotone, strictly decreasing function such that $h(1) > 0$, $h(N) = 0$.*

This assumption implies

$$\underline{h} \leq h(M) \quad \text{if} \quad 0 < M < N, \qquad \text{and} \qquad h(M) \leq \bar{h} \quad \text{if} \quad 0 < M \leq N, \tag{3}$$

for $M \in \mathbb{N}$ and $\underline{h} = h(N-1)$ and $\bar{h} = h(1)$. One possible choice for h is $h(M) = (N-M)/N$, $0 < M \leq N$.

Suppose ϕ_i, $1 \leq i \leq N$, be continuously differentiable and let $\|\cdot\|$ denote the 2-norm.

The method introduced in this section combines the Inexact Restoration, an approach for optimization of functions evaluated inexactly, with the trust-region methods. We will refer to it as IRETR. It employs the merit function

$$\Psi(x, M, \theta) = \theta f_M(x) + (1-\theta)h(M), \tag{4}$$

with $\theta \in (0,1)$ and aims to minimize both f_M and the infeasibility h. Since the reductions in the values of f_M and h may not be achieved simultaneously, a weight θ is used and a trust-region method is employed to generate a sequence $\{(x_k, N_k, \theta_k)\}$ such that $\Psi(x_k, N_k, \theta_k) < \Psi(x_{k-1}, N_{k-1}, \theta_k)$. The main theoretical properties of the new method, shown in the next section, are: the sequence $\{\theta_k\}$ is nonicreasing and uniformly bounded away from zero, $N_k = N$ for all k sufficiently large and $\|\nabla f_N(x_k)\| \to 0$ as $k \to \infty$.

Concerning the trust-region problem, suppose that x_k is given. Then, a trial sample size N_{k+1} is selected, $I_{N_{k+1}} \subseteq \{1, \dots, N\}$ is chosen and the model $m_k(p)$ for $f_{N_{k+1}}$ around x_k of the form

$$m_k(p) = f_{N_{k+1}}(x_k) + \nabla f_{N_{k+1}}(x_k)^T p + \frac{1}{2} p^T B_{k+1} p, \tag{5}$$

is built. Here $\nabla f_{N_{k+1}}$ denotes the gradient of $f_{N_{k+1}}$ and $B_{k+1} \in \mathbb{R}^{n \times n}$ is a symmetric approximation to the Hessian $\nabla^2 f_{N_{k+1}}(x_k)$ in case ϕ_i, $1 \le i \le N$, are twice continuously differentiable. Trivially $m_k(0) = f_{N_{k+1}}(x_k)$ and the smaller $h(N_{k+1})$, the larger becomes the accuracy in the approximation to f_N and ∇f_N. Then, letting $\Delta_k > 0$ denote the trust-region radius and $\mathcal{B}_k = \{x_k + p \in \mathbb{R}^n : \|p\| \le \Delta_k\}$ be the trust-region, the trust-region problem is

$$\min_{\|p\| \le \Delta_k} m_k(p). \tag{6}$$

As in the standard trust-region schemes, problem (6) is solved approximately and the computed step p_k is required to provide a sufficient reduction in the model in terms of the Cauchy step p_k^C, i.e., the minimizer of the model m_k along the steepest descent $-\nabla f_{N_{k+1}}(x_k)$ within $\mathcal{B}_k$

$$p_k^C = \underset{\substack{p = -t\nabla f_{N_{k+1}}(x_k),\ t>0 \\ \|p\| \le \Delta_k}}{\operatorname{argmin}} m_k(p). \tag{7}$$

Then, if a sufficient reduction in the function Ψ is achieved, the step p_k is accepted and the new iterate x_{k+1} is set equal to $x_k + p_k$. Otherwise, the step is rejected and the trust-region radius is reduced. The specific form of the predicted and actual reduction used in the acceptance criterion will be given below, after detailing the Algorithm's steps.

Now we present the new Algorithm IRETR which aims at finding an ε_g–accurate first-order optimality point defined as follows

$$\|\nabla f_{N_{k+1}}(x_k)\| \leq \varepsilon_g \quad \text{and} \quad N_k = N, \tag{8}$$

and comment on it, see Algorithm 1.

Given x_k, N_k and θ_k we describe the kth iteration. In Step 1 the feasibility is improved. If $N_k < N$, we predict the cardinality $\widetilde{N}_{k+1}$ such that the value $h(\widetilde{N}_{k+1})$ is smaller than $h(N_k)$ and at most equal to a prefixed fraction of $h(N_k)$. In case $h(M) = (N - M)/N,\ 0 < M \leq N$, taking into account that N_k and $\widetilde{N}_{k+1}$ are integers it can be shown that condition (9) holds if and only if $0 < N_k < \widetilde{N}_{k+1}$ provided that $h(2)/h(1) < r < 1$.

In Step 2, an attempt is made to reduce the computational effort i.e. to enlarge infesibility; N_{k+1} is chosen such that $N_{k+1} \leq \widetilde{N}_{k+1}$ and the bounded deterioration (10) on the value of $h(N_{k+1})$ with respect to $h(\widetilde{N}_{k+1})$ is imposed. In principal such control allows us to reduce N_{k+1} below both N_k and $\widetilde{N}_{k+1}$. On the other hand, the upper bound in (10) depends on the trust-region radius and N_{k+1} will be equal to $\widetilde{N}_{k+1}$ whenever Δ_k is small enough. If $N_k = N$, the stopping criterion $\|\nabla f_{N_{k+1}}(x_k)\| \leq \varepsilon_g$ is checked. This is supported by the fact that, when $N_k = N$, we may expect N_{k+1} be close to N and $\nabla f_{N_{k+1}}(x_k)$ be close to $\nabla f_N(x_k)$ in a probabilistic sense; we will further discuss this issue in Section 3. If (8) is not met, using $I_{N_{k+1}} \subseteq \{1, \ldots, N\}$, the trust-region model $m_k(p)$ is built and (6) is approximately solved. The computed step p_k is required to provide the sufficient reduction (11) in the model in terms of the Cauchy step p_k^C.

The acceptance rule for p_k in Step 5 depends on the predicted and actual reduction defined as follows:

$$\begin{aligned}
\mathrm{Pred}_k(\theta) &= \theta(f_{N_k}(x_k) - m_k(p_k)) + (1-\theta)(h(N_k) - h(\widetilde{N}_{k+1})), && (15)\\
\mathrm{Ared}_k(\theta) &= \Psi(x_k, N_k, \theta) - \Psi(x_k + p_k, N_{k+1}, \theta) \\
&= \theta(f_{N_k}(x_k) - f_{N_{k+1}}(x_k + p_k)) + (1-\theta)(h(N_k) - h(N_{k+1})), && (16)
\end{aligned}$$

where the last equality follows from (4). We observe that Pred_k uses the last accepted values $f_{N_k}(x_k)$ and N_k and is a linear combination of two predicted values: the predicted model decrease $f_{N_k}(x_k) - m_k(p_k)$ and the predicted infeasibility decrease $h(N_k) - h(\widetilde{N}_{k+1})$. As for Ared_k, given θ, it measures the actual reduction of Ψ.

Algorithm 1 The algorithm IRETR

```
Given $x_0 \in \mathbb{R}^n$, $N_0$ integer in $(0, N]$, $\theta_0 \in (0,1)$, $B_0 \in \mathbb{R}^{n\times n}$, $\Delta_0 > 0$,
$0 < \zeta_1 < 1 < \zeta_2$, $\gamma \in (0,1]$, $r, \eta, \tau \in (0,1)$, $\mu \in [0,1)$ the accuracy level
$\varepsilon_g \geq 0$.

0. Set $k = 0$, $\mathcal{T}_0 = 0$, $\Delta_0 = \Delta_0^{(\mathcal{T}_0)}$;
1. If $N_k < N$, find $\widetilde{N}_{k+1}$ such that $N_k < \widetilde{N}_{k+1} \leq N$, and
       $h(\widetilde{N}_{k+1}) \leq r h(N_k). \qquad (9)$
   If $N_k = N$, set $\widetilde{N}_{k+1} = N$.
2. Choose $N_{k+1}$ such that $N_{k+1} \leq \widetilde{N}_{k+1}$, and
       $h(N_{k+1}) - h(\widetilde{N}_{k+1}) \;\; \leq \;\; \mu \left(\Delta_k^{(\mathcal{T}_k)}\right)^{1+\gamma}. \qquad (10)$
   If $N_k = N$ and $\|\nabla f_{N_{k+1}}(x_k)\| \leq \varepsilon_g$, stop.
   Build the model $m_k(p)$ for $f_{N_{k+1}}(x_k)$ in (5).
   Find an approximate trust-region solution $p_k$ such that
       $m_k(0) - m_k(p_k) \;\; \geq \;\; \tau(m_k(0) - m_k(p_k^C)) \qquad (11)$
   where $p_k^C$ is given in (7).
3. If $N_k = N$ and $N_{k+1} < N$ and
       $f_N(x_k) - m_k(p_k) \;\; < \;\; \tau(m_k(0) - m_k(p_k^C)) \qquad (12)$
    take $\Delta_k^{(\mathcal{T}_k+1)} = \zeta_1 \Delta_k^{(\mathcal{T}_k)}$, set $\mathcal{T}_k = \mathcal{T}_k + 1$ and repeat Step 2.
4. Compute the penalty parameter $\theta_{k+1}$
       $\theta_{k+1} = \begin{cases} \theta_k & \text{if } \operatorname{Pred}_k(\theta_k) \geq \eta(h(N_k) - h(\widetilde{N}_{k+1})) \\ \dfrac{(1-\eta)(h(N_k) - h(\widetilde{N}_{k+1}))}{m_k(p_k) - f_{N_k}(x_k) + h(N_k) - h(\widetilde{N}_{k+1})} & \text{otherwise.} \end{cases} \qquad (13)$
5. If
       $\operatorname{Ared}_k(\theta_{k+1}) \geq \eta \operatorname{Pred}_k(\theta_{k+1}), \qquad (14)$
   Set $x_{k+1} = x_k + p_k$, $\Delta_k = \Delta_k^{(\mathcal{T}_k)}$. Choose $\Delta_{k+1}^{(0)} \in [\Delta_k, \zeta_2 \Delta_k]$,
   set $k = k+1, \mathcal{T}_k = 0$, and go to Step 1.
   Else take $\Delta_k^{(\mathcal{T}_k+1)} = \zeta_1 \Delta_k^{(\mathcal{T}_k)}$, set $\mathcal{T}_k = \mathcal{T}_k + 1$ and go to Step 2.
```

The new penalty parameter θ_{k+1} computed in Step 4 is the largest value that ensures

$$\mathrm{Pred}_k(\theta_{k+1}) \geq \eta(h(N_k) - h(\widetilde{N}_{k+1})) \geq 0, \tag{17}$$

as $h(N_k) - h(\widetilde{N}_{k+1}) \geq 0$ by (9). In case $N_k < \widetilde{N}_{k+1}$ such condition implies $\mathrm{Pred}_k(\theta_{k+1})$ strictly positive. In case $N_k = \widetilde{N}_{k+1} = N$, $\mathrm{Pred}_k(\theta)$ reduces to $\theta(f_N(x_k) - m_k(p_k))$ and from (11) it follows $\mathrm{Pred}_k(\theta) \geq \tau\theta(m_k(0) - m_k(p_k^C)) > 0$ whenever $N_{k+1} = N$. On the other hand, in case $N_k = \widetilde{N}_{k+1} = N$ and $N_{k+1} < N$, Step 3 is necessary to enforce positivity of $\mathrm{Pred}_k(\theta_{k+1})$ as $m_k(0) = f_{N_{k+1}}(x_k) \neq f_N(x_k)$. In fact, $\mathrm{Pred}_k(\theta) > 0$ follows from taking a step such that $f_N(x_k) - m_k(p_k) \geq \tau(m_k(0) - m_k(p_k^C))$. We further notice that attempting $N_{k+1} < N$ when $N_k = N$ is meaningful if the model is a good approximation of f_N around x_k and thus one can expect some progress, or at least a limited deterioration in the value of the full objective function f_N. Enforcing $f_N(x_k) - m_k(p_k) \geq \tau(m_k(0) - m_k(p_k^C))$ is a minimal requirement on the agreement between f_N at x_k and the model at the trial step.

Finally, in Step 5 the step p_k is accepted if the ratio between the predicted reduction $\mathrm{Pred}_k(\theta_{k+1})$ and the actual reduction $Ared_k(\theta_{k+1})$ is larger than a prefixed scalar η, otherwise the trust-region radius is reduced and the procedure is repeated starting from Step 2.

Notice that the trust-region size can be reduced several times during one iteration, i.e., only successful iterations yield to the increment of the iteration counter k. To emphasize this fact, within each iteration, we introduce an additional counter $\mathcal{T}_k$ for the number of decreases of the trust-region size. The feasibility measure N_{k+1} might be modified several times within one iteration as well, but changes due to (10) and (12) do not necessarily correspond to the number of reductions of the trust-region size. The penalty parameter θ_k has an analogous behaviour. For this reason and to avoid notation clustering, we do not introduce additional counters for N_{k+1} and θ_{k+1} within the same iteration.

We start the analysis of the new method proving that the kth iteration of Algorithm IRETR is well defined since appropriate values of N_{k+1} and θ_{k+1} will be reached in a finite number of attempts. Here and in Section 5, B_{k+1} can be the null matrix and our analysis covers the use of both first-order and second-order models.

Lemma 2.1 *Steps 2 and 3 of Algorithm* IRETR *are well-defined.*

Proof. For any positive $\Delta_k^{(\mathcal{T}_k)}$ inequality (10) trivially holds in the limit case $N_{k+1} = \widetilde{N}_{k+1}$. Analogously, Step 3 can not be repeated infinitely many times as for $\mathcal{T}_k$ large enough, $\Delta_k^{(\mathcal{T}_k)}$ will be small enough to yield $N_{k+1} = \widetilde{N}_{k+1} = N$. □

We now make the following assumption.

Assumption 2.2 $\{x_k\} \subset \Omega$ *where* Ω *is a compact set in* $\mathbb{R}^n$.

Lemma 2.2 *Let Assumptions 2.1 and 2.2 hold. Suppose that* ϕ_i, $1 \le i \le N$, *are continuous in* Ω. *Then the sequence* $\{\theta_k\}$ *built in Algorithm* IRETR *is positive, nonincreasing and bounded away from zero,* $\theta_{k+1} \ge \underline{\theta} > 0$ *with* $\underline{\theta}$ *independent of* k *and* (17) *holds.*

Proof. We have $\theta_0 > 0$ and proceed by induction assuming that θ_k is positive. First consider the case where $N_k = \widetilde{N}_{k+1}$ (equivalently $N_k = \widetilde{N}_{k+1} = N$). Then $h(N_k) - h(\widetilde{N}_{k+1}) = 0$ and, due to Step 3, $\text{Pred}_k(\theta) = \theta(f_{N_k}(x_k) - m_k(p_k)) > 0$ for any positive θ. Thus $\theta_{k+1} = \theta_k$ and (17) holds.

Let now suppose $N_k < \widetilde{N}_{k+1}$. If inequality $\text{Pred}_k(\theta_k) \ge \eta(h(N_k) - h(\widetilde{N}_{k+1}))$ holds then $\theta_{k+1} = \theta_k$ satisfies (17). Otherwise, we have

$$\theta_k(f_{N_k}(x_k) - m_k(p_k) - (h(N_k) - h(\widetilde{N}_{k+1}))) < (\eta - 1)(h(N_k) - h(\widetilde{N}_{k+1})),$$

and since the right hand-side is negative by construction, it follows

$$f_{N_k}(x_k) - m_k(p_k) - (h(N_k) - h(\widetilde{N}_{k+1})) < 0.$$

Consequently, $\text{Pred}_k(\theta) \ge \eta(h(N_k) - h(\widetilde{N}_{k+1}))$ is satisfied if

$$\theta(f_{N_k}(x_k) - m_k(p_k) - (h(N_k) - h(\widetilde{N}_{k+1}))) \ge (\eta - 1)(h(N_k) - h(\widetilde{N}_{k+1})),$$

i.e., if

$$\theta \le \theta_{k+1} \overset{\text{def}}{=} \frac{(1-\eta)(h(N_k) - h(\widetilde{N}_{k+1}))}{m_k(p_k) - f_{N_k}(x_k) + h(N_k) - h(\widetilde{N}_{k+1})}.$$

Hence θ_{k+1} is the largest value satisfying (17) and $\theta_{k+1} < \theta_k$.

Let us now prove that $\theta_{k+1} \ge \underline{\theta}$. Using Assumptions 2.2 and continuity of ϕ_i, $1 \le i \le N$, let

$$\kappa_\phi = \max_{\substack{1 \le i \le N \\ x \in \Omega}} |\phi_i(x)|. \tag{18}$$

Then, using (3), for M such that $0 < M \le N$ there holds

$$\begin{aligned} f_N(x_k) - f_M(x_k) &= \frac{1}{N}\sum_{i \in I_N} \phi_i(x_k) - \frac{1}{M}\sum_{i \in I_M} \phi_i(x_k) \\ &= \left(\frac{1}{N} - \frac{1}{M}\right)\sum_{i \in I_M} \phi_i(x_k) + \frac{1}{N}\sum_{i \in I_N \setminus I_M} \phi_i(x_k), \end{aligned}$$

and therefore for any integer M, $0 < M \leq N$

$$\begin{aligned} |f_N(x_k) - f_M(x_k)| &\leq \frac{N-M}{NM} M\kappa_\phi + \frac{N-M}{N}\kappa_\phi \\ &= \frac{2(N-M)\kappa_\phi}{N\,h(M)} h(M) \\ &\leq \frac{2(N-M)\kappa_\phi}{N\,\underline{h}} h(M) \\ &\leq \frac{2(N-1)\kappa_\phi}{N\,\underline{h}} h(M) \\ &\overset{\text{def}}{=} \sigma h(M). \qquad (19) \end{aligned}$$

Also note that by (9) and (3)

$$h(N_k) - h(\widetilde{N}_{k+1}) \geq (1-r)h(N_k) \geq (1-r)\underline{h}. \qquad (20)$$

Moreover,

$$\begin{aligned} & m_k(p_k) - f_{N_k}(x_k) + h(N_k) - h(\widetilde{N}_{k+1}) \leq m_k(p_k) - f_{N_k}(x_k) + h(N_k) \\ & \leq m_k(0) - f_{N_k}(x_k) + \bar{h} = f_{N_{k+1}}(x_k) - f_{N_k}(x_k) + \bar{h} \\ & \leq |f_{N_{k+1}}(x_k) - f_N(x_k)| + |f_N(x_k) - f_{N_k}(x_k)| + \bar{h} \\ & \leq \sigma(h(N_k) + h(N_{k+1})) + \bar{h} \leq (2\sigma+1)\bar{h}, \end{aligned}$$

and θ_{k+1} in (13) satisfies

$$\theta_{k+1} \geq \frac{(1-\eta)(1-r)\underline{h}}{(2\sigma+1)\bar{h}} \overset{\text{def}}{=} \underline{\theta},$$

and the proof is completed. □

To establish the well-definiteness of Steps 4 and 5, we make the following assumptions.

Assumption 2.3 *The gradients $\nabla\phi_i$, $1 \leq i \leq N$, are Lipschitz continuous on the segments $[x_k, x_k + p_k]$, for all $k \geq 0$ and for all p_k generated in the repetition of Steps 2–5.*

Assumption 2.4 *There exists positive κ_B such that for all k*

$$\|B_{k+1}\| \leq \kappa_B.$$

By Assumption 2.3 there is a $t \in (0,1)$ such that

$$f_{N_{k+1}}(x_k+p_k)-m_k(p_k) = \int_0^1 \left(\nabla f_{N_{k+1}}(x_k+tp_k) - \nabla f_{N_{k+1}}(x_k)\right)^T p_k dt - \frac{1}{2} p_k^T B_{k+1} p_k,$$

[18, Lemma 4.1.2]. Consequently, using Assumptions 2.2–2.4 we have

$$|f_{N_{k+1}}(x_k+p_k) - m_k(p_k)| \le \kappa_T \Delta_k^2, \tag{21}$$

with $\kappa_T = (L + \kappa_B/2)$ and L depending on the Lipschitz constants of $\nabla \phi_i$, $1 \le i \le N$.

In the next result we use the key inequality

$$m_k(0) - m_k(p_k^C) \ge \frac{1}{2} \|\nabla f_{N_{k+1}}(x_k)\| \min\left\{ \frac{\|\nabla f_{N_{k+1}}(x_k)\|}{\beta}, \Delta_k \right\}, \tag{22}$$

with $\beta = 1 + \kappa_B$, see [16, Theorem 6.3.1].

Lemma 2.3 *Let Assumptions 2.1– 2.4 hold. Assume $\theta_k \in (0,1)$ and θ_{k+1} as in (13). Then, Steps 4 and 5 of Algorithm* IRETR *are well defined.*

Proof. Let us prove that $\mathrm{Ared}_k(\theta_{k+1}) - \eta \mathrm{Pred}_k(\theta_{k+1})$ is strictly positive if $\Delta_k^{(\mathcal{T}_k)}$ is small enough, i.e., after a finite number $\mathcal{T}_k$ of reductions of the trust-region radius. Let θ_{k+1} be computed at Step 4 for some $\Delta_k^{(\mathcal{T}_k)}$. By (15) and (16), we have

$$\begin{aligned}
&\mathrm{Ared}_k(\theta_{k+1}) - \eta \mathrm{Pred}_k(\theta_{k+1}) \\
&= \theta_{k+1}(f_{N_k}(x_k) - f_{N_{k+1}}(x_k+p_k)) + (1-\theta_{k+1})(h(N_k) - h(N_{k+1})) - \\
&\quad \eta\theta_{k+1}(f_{N_k}(x_k) - m_k(p_k)) - \eta(1-\theta_{k+1})(h(N_k) - h(\widetilde{N}_{k+1})) \\
&= \theta_{k+1}(f_{N_k}(x_k) - m_k(p_k)) + \theta_{k+1}(m_k(p_k) - f_{N_{k+1}}(x_k+p_k)) + \\
&\quad (1-\theta_{k+1})(h(N_k) - h(\widetilde{N}_{k+1})) + (1-\theta_{k+1})(h(\widetilde{N}_{k+1}) - h(N_{k+1})) - \\
&\quad \eta\theta_{k+1}(f_{N_k}(x_k) - m_k(p_k)) - \eta(1-\theta_{k+1})(h(N_k) - h(\widetilde{N}_{k+1})) \\
&= (1-\eta)\left(\theta_{k+1}(f_{N_k}(x_k) - m_k(p_k)) + (1-\theta_{k+1})(h(N_k) - h(\widetilde{N}_{k+1}))\right) + \\
&\quad \theta_{k+1}(m_k(p_k) - f_{N_{k+1}}(x_k+p_k)) + (1-\theta_{k+1})(h(\widetilde{N}_{k+1}) - h(N_{k+1})) \\
&= (1-\eta)\mathrm{Pred}_k(\theta_{k+1}) + \theta_{k+1}(m_k(p_k) - f_{N_{k+1}}(x_k+p_k)) + \\
&\quad (1-\theta_{k+1})(h(\widetilde{N}_{k+1}) - h(N_{k+1})).
\end{aligned}$$

We now distinguish three cases.

i) If $h(N_k) - h(\widetilde{N}_{k+1}) > 0$ then using (17) we get

$$\begin{aligned} \mathrm{Ared}_k(\theta_{k+1}) - \eta \mathrm{Pred}_k(\theta_{k+1}) \quad \geq \quad & \eta(1-\eta)(h(N_k) - h(\widetilde{N}_{k+1})) \\ & + \theta_{k+1}(m_k(p_k) - f_{N_{k+1}}(x_k + p_k)) \\ & + (1 - \theta_{k+1})(h(\widetilde{N}_{k+1}) - h(N_{k+1})). \end{aligned} \tag{23}$$

The first term in the above right hand-side is strictly positive and uniformly bounded from below due to (20). On the other hand, by (21) and (10)

$$|\theta_{k+1}(m_k(p_k) - f_{N_{k+1}}(x_k + p_k)) + (1 - \theta_{k+1})(h(\widetilde{N}_{k+1}) - h(N_{k+1}))| \leq \kappa_T \left(\Delta_k^{(\mathcal{T}_k)}\right)^2 + \mu \left(\Delta_k^{(\mathcal{T}_k)}\right)^{1+\gamma} \tag{24}$$

Therefore, for $\Delta_k^{\mathcal{T}_k}$ small enough we have $\mathrm{Ared}_k(\theta_{k+1}) - \eta \mathrm{Pred}_k(\theta_{k+1}) > 0$ and the iteration finishes.

ii) If $h(N_k) - h(\widetilde{N}_{k+1}) = 0$ (equivalently $N_k = \widetilde{N}_{k+1} = N$) and $N_{k+1} = N$ then using (15) and (16) we have

$$\begin{aligned} \mathrm{Ared}_k(\theta_{k+1}) - \eta \mathrm{Pred}_k(\theta_{k+1}) \quad = \quad & (1-\eta)\theta_{k+1}(f_N(x_k) - m_k(p_k)) + \\ & \theta_{k+1}(m_k(p_k) - f_N(x_k + p_k)). \end{aligned}$$

Thus, by (11), (21) and (22), if $\Delta_k^{(\mathcal{T}_k)}$ is small enough we get

$$\begin{aligned} & \mathrm{Ared}_k(\theta_{k+1}) - \eta \mathrm{Pred}_k(\theta_{k+1}) \\ & \quad \geq \tau(1-\eta)\theta_{k+1}(m_k(0) - m_k(p_k^C)) + \theta_{k+1}(m_k(p_k) - f_N(x_k + p_k)) \\ & \quad \geq \frac{1}{2}\tau(1-\eta)\theta_{k+1}\|\nabla f_N(x_k)\|\Delta_k^{(\mathcal{T}_k)} - \theta_{k+1}|m_k(p_k) - f_N(x_k + p_k)| \\ & \quad \geq \frac{1}{2}\tau\underline{\theta}(1-\eta)\|\nabla f_N(x_k)\|\Delta_k^{(\mathcal{T}_k)} - |m_k(p_k) - f_N(x_k + p_k)| \\ & \quad \geq \left(\frac{1}{2}\tau\underline{\theta}(1-\eta)\|\nabla f_N(x_k)\| - \kappa_T \Delta_k^{(\mathcal{T}_k)}\right)\Delta_k^{(\mathcal{T}_k)}, \end{aligned} \tag{25}$$

and the last bound is positive for some finite $\mathcal{T}_k$.

iii) Finally, suppose $h(N_k) - h(\widetilde{N}_{k+1}) = 0$ (equivalently $N_k = \widetilde{N}_{k+1} = N$) and $N_{k+1} < N$ then using (15) and (16) we have

$$\begin{aligned} \mathrm{Ared}_k(\theta_{k+1}) - \eta \mathrm{Pred}_k(\theta_{k+1}) = (1-\eta)\theta_{k+1}(f_N(x_k) - m_k(p_k)) + \\ \theta_{k+1}(m_k(p_k) - f_{N_{k+1}}(x_k + p_k)) - (1 - \theta_{k+1})h(N_{k+1}). \end{aligned}$$

Thus, by Step 3 of Algorithm 2.1, (21) and (22), if $\Delta_k^{\mathcal{T}_k}$ is small enough we get

$$
\begin{aligned}
&\text{Ared}_k(\theta_{k+1}) - \eta \text{Pred}_k(\theta_{k+1}) \geq && (26)\\
&\quad \geq (1-\eta)\underline{\theta}\tau(m_k(0) - m_k(p_k^C)) - \theta_{k+1}|m_k(p_k) - f_{N_{k+1}}(x_k + p_k)| - h(N_{k+1})\\
&\quad \geq \frac{1}{2}\tau\underline{\theta}(1-\eta)\|\nabla f_{N_{k+1}}(x_k)\|\Delta_k^{(\mathcal{T}_k)} - \theta_{k+1}|m_k(p_k) - f_{N_{k+1}}(x_k + p_k)| - h(N_{k+1})\\
&\quad \geq \left(\frac{1}{2}\tau\underline{\theta}(1-\eta)\|\nabla f_{N_{k+1}}(x_k)\| - \kappa_T \Delta_k^{(\mathcal{T}_k)} - \mu\left(\Delta_k^{(\mathcal{T}_k)}\right)^\gamma\right)\Delta_k^{(\mathcal{T}_k)}, && (27)
\end{aligned}
$$

and the last bound is positive for some finite $\mathcal{T}_k$. □

The analysis presented in the rest of this section concerns the case where Algorithm IRETR is invoked with $\varepsilon_g = 0$ and does not terminate in a finite number of steps. Each iteration $k-1$ of the Algorithm ends up with the accepted iterate $x_k = x_{k-1} + p_{k-1}$ and the final sample size N_k. In the following statements we are going to prove that $h(N_k) \to 0$ and therefore the full sample is eventually reached and maintained.

Theorem 2.4 *Let Assumptions 2.1–2.4 hold. Then* $h(N_k) \to 0$.

Proof. Inequalities (9) and (17) imply

$$
h(N_k) \leq \frac{h(N_k) - h(\widetilde{N}_{k+1})}{1-r} \leq \frac{\text{Pred}_k(\theta_{k+1})}{\eta(1-r)}. \qquad (28)
$$

We prove by contradiction that $\lim_{k\to\infty} \text{Pred}_k(\theta_{k+1}) = 0$.

Taking into account that at termination of iteration k we have $x_{k+1} = x_k + p_k$ and $\text{Ared}_k(\theta_{k+1}) \geq \eta \text{Pred}_k(\theta_{k+1})$, using (14) and (16) we have

$$
\begin{aligned}
\theta_{k+2} f_{N_{k+1}}(x_{k+1}) &\leq \theta_{k+2} f_{N_{k+1}}(x_{k+1}) + \text{Ared}_k(\theta_{k+1}) - \eta \text{Pred}_k(\theta_{k+1})\\
&= \theta_{k+1} f_{N_k}(x_k) + (\theta_{k+2} - \theta_{k+1}) f_{N_{k+1}}(x_{k+1}) +\\
&\quad (1-\theta_{k+1})(h(N_k) - h(N_{k+1})) - \eta \text{Pred}_k(\theta_{k+1})\\
&\leq \theta_{k+1} f_{N_k}(x_k) + (\theta_{k+1} - \theta_{k+2}) \max_{x\in\Omega} \sum_{i\in I_{N_{k+1}}} |\phi_i(x)| +\\
&\quad (1-\theta_{k+1})(h(N_k) - h(N_{k+1})) - \eta \text{Pred}_k(\theta_{k+1}).
\end{aligned}
$$

Using (18) we can rewrite the above inequality as

$$
\begin{aligned}
\theta_{k+2} f_{N_{k+1}}(x_{k+1}) &\leq \theta_{k+1} f_{N_k}(x_k) + (\theta_{k+1} - \theta_{k+2}) N\kappa_\phi\\
&+ (1-\theta_{k+1})(h(N_k) - h(N_{k+1})) - \eta \text{Pred}_k(\theta_{k+1}).
\end{aligned}
$$

Then using recurrence, and $-(1-\theta_{k+1})h(N_{k+1}) \leq 0$ we get

$$\begin{aligned}
\theta_{k+2} f_{N_{k+1}}(x_{k+1}) &\leq \theta_k f_{N_{k-1}}(x_{k-1}) + (\theta_k - \theta_{k+2})N\kappa_\phi + (1-\theta_k)(h(N_{k-1}) - h(N_k)) \\
&\quad +(1-\theta_{k+1})(h(N_k) - h(N_{k+1}) - \eta \sum_{j=k-1}^{k} \mathrm{Pred}_j(\theta_{j+1}) \\
&\leq \theta_k f_{N_{k-1}}(x_{k-1}) + (\theta_k - \theta_{k+2})N\kappa_\phi + \\
&\quad (1-\theta_k)h(N_{k-1}) + (\theta_k - \theta_{k+1})h(N_k) - \eta \sum_{j=k-1}^{k} \mathrm{Pred}_j(\theta_{j+1}).
\end{aligned}$$

Repeating this argument, using $(\theta_j - \theta_{j+1}) \geq 0$ from Lemma 2.2 and (3) we obtain

$$\begin{aligned}
\theta_{k+2} f_{N_{k+1}}(x_{k+1}) &\leq \theta_1 f_{N_0}(x_0) + (\theta_1 - \theta_{k+2})N\kappa_\phi + (1-\theta_1)h(N_0) + \\
&\quad \sum_{j=1}^{k} (\theta_j - \theta_{j+1})h(N_j) - \eta \sum_{j=0}^{k} \mathrm{Pred}_j(\theta_{j+1}) \\
&\leq \theta_1 f_{N_0}(x_0) + (1-\underline{\theta})N\kappa_\phi + (1-\underline{\theta})\bar{h} + \sum_{j=1}^{k} (\theta_j - \theta_{j+1})\bar{h} - \\
&\quad \eta \sum_{j=0}^{k} \mathrm{Pred}_j(\theta_{j+1}) \\
&\leq \theta_1 f_{N_0}(x_0) + (1-\underline{\theta})N\kappa_\phi + (1-\underline{\theta})\bar{h} + (\theta_1 - \theta_{k+1})\bar{h} - \\
&\quad \eta \sum_{j=0}^{k} \mathrm{Pred}_j(\theta_{j+1}) \\
&\leq \theta_1 f_{N_0}(x_0) + (1-\underline{\theta})N\kappa_\phi + 2(1-\underline{\theta})\bar{h} - \eta \sum_{j=0}^{k} \mathrm{Pred}_j(\theta_{j+1}).
\end{aligned}$$

By (19) and (3) we have

$$\begin{aligned}
\theta_{k+2} f_{N_{k+1}}(x_{k+1}) &= \theta_{k+2} f_N(x_{k+1}) + \theta_{k+2}(f_{N_{k+1}}(x_{k+1}) - f_N(x_{k+1})) \\
&\geq \theta_{k+2} f_N(x_{k+1}) - \theta_{k+2}|(f_{N_{k+1}}(x_{k+1}) - f_N(x_{k+1}))| \\
&\geq \theta_{k+2} f_N(x_{k+1}) - \sigma\bar{h},
\end{aligned}$$

and therefore

$$\theta_{k+2} f_N(x_{k+1}) \leq \xi - \eta \sum_{j=0}^{k} \mathrm{Pred}_j(\theta_{j+1}), \tag{29}$$

where

$$\xi = \theta_1 f_{N_0}(x_0) + (1-\underline{\theta})N\kappa_\phi + 2(1-\underline{\theta}+\sigma)\bar{h}, \tag{30}$$

is independent of k.

Noting that $\mathrm{Pred}_j(\theta_{j+1}) \geq 0$, we can conclude that if $\mathrm{Pred}_j(\theta_{j+1})$ is not tending to zero, then $\sum_{j=0}^{\infty} \mathrm{Pred}_j(\theta_{j+1})$ is diverging and this implies that f_N is unbounded below in Ω. This contradicts the compactness of Ω. □

Corollary 2.5 *Let Assumptions 2.1–2.4 hold. Then $N_k = N$ for all k sufficiently large.*

Proof. By Theorem 2.4 and Assumption 2.1, it follows $h(N_k) < h(N-1)$ for all k sufficiently large. This implies $N_k = N$. □

Corollary 2.6 *Let Assumptions 2.1–2.4 hold. Then, for k sufficiently large, the iterations are generated by a (standard) trust-region scheme on f_N and*

i) $\mathrm{liminf}_{k\to\infty} \|\nabla f_N(x_k)\| = 0.$

ii) $\lim_{k\to\infty} \|\nabla f_N(x_k)\| = 0$*, provided that f_N is Lipschitz continuous in Ω.*

Proof. By Corollary 2.5 we know that at termination of iteration $k-1$ we have $N_k = N$ for all k sufficiently large. Thus eventually, $x_{k+1} = x_k + p_k$ with p_k satisfying (14) which now takes the form of the standard acceptance rule of the trial point in trust-region methods, i.e,

$$\frac{f_N(x_{k+1}) - f_N(x_k)}{m_k(0) - m_k(p_k)} \geq \eta.$$

As a consequence, Theorem 4.6 in [32] yields item *i*). Item *ii*) is guaranteed by [32, Theorem 4.7]. □

3 On the realization of the algorithm

The realization of Algorithm IRETR raises many issues and in this section we discuss two important aspects: the form of the model used and related properties, and a computationally convenient adaptation of the rule for choosing N_{k+1} eventually. We will further address implementation issues in Section 6.

Various models of the form (5) can be built. One possibility is the linear model

$$m_k(p) = f_{N_{k+1}}(x_k) + \nabla f_{N_{k+1}}(x_k)^T p,$$

which gives rise to a gradient method and step p_k

$$p_k = -\Delta_k \frac{\nabla f_{N_{k+1}}(x_k)}{\|\nabla f_{N_{k+1}}(x_k)\|}.$$

Namely, Algorithm IRETR becomes a subsampled gradient method with variable stepsize determined accordingly to the trust-region strategy.

Another possibility is to use quadratic models of the form

$$m_k(p) = f_{N_{k+1}}(x_k) + \nabla f_{N_{k+1}}(x_k)^T p + \frac{1}{2} p^T B_{k+1} p,$$

and fully exploit the advantages of the trust-region framework. If all functions ϕ_i are twice continuously differentiable one can build the quadratic model

$$m_k(p) = f_{N_{k+1}}(x_k) + \nabla f_{N_{k+1}}(x_k)^T p + \frac{1}{2} p^T \nabla^2 f_{D_{k+1}}(x_k) p,$$

with $1 \le D_{k+1} \le N_{k+1}$ and $I_{D_{k+1}} \subseteq I_{N_{k+1}}$. In fact, the Hessian matrix $\nabla^2 f_{N_{k+1}}(x)$ is approximated via subsampling by

$$B_{k+1} = \frac{1}{D_{k+1}} \sum_{i \in I_{D_{k+1}}} \nabla^2 \phi_i(x_k). \tag{31}$$

The cardinality of $I_{D_{k+1}}$ now controls the precision of Hessian approximation and allows for trade-off between precision and computational costs. This particular form of Hessian approximation will be analysed in details for strongly convex functions in the next section.

The use of quadratic models is crucial for the computation of $(\varepsilon_g, \varepsilon_H)$-approximate second order critical point of nonconvex problems (1), i.e., a point x such that

$$\|\nabla f_N(x)\| \le \varepsilon_g, \quad \lambda_{\min}(\nabla^2 f_{D_{k+1}}(x)) \ge -\varepsilon_H, \tag{32}$$

Supposing that full precision is reached, $N_k = N$, the trust-region problem (6) has to be solved approximately finding p_k such that

$$m_k(p_k) \le m_k(p_k^C) \text{ and } m_k(p_k) \le m_k(p_k^E) \text{ if } \lambda_{\min}(\nabla^2 f_{D_{k+1}}(x_k)) < 0, \tag{33}$$

where p_k^C is the Cauchy point (7) and p_k^E is a negative curvature direction such that $(p_k^E)^T \nabla^2 f_{D_{k+1}}(x_k) p_k^E \le \upsilon \lambda_{\min}(\nabla^2 f_{D_{k+1}}(x_k)) \|p_k^E\|^2$ for some $\upsilon \in (0,1]$, [16, §6.6].

We refer to [38, Theorem 1] for results on the computation of approximated second-order optimal solutions using trust-region methods with full function and gradient and subsampled Hessian.

Let us now address the choice of the stopping criterion in Algorithm IRETR. Notice that the Algorithm may stop even if full precision at iteration k is not achieved (i.e. $N_{k+1} < N$), provided that $N_k = N$. This choice is supported by observing that suitable sample sizes provide an accurate approximation $\nabla f_{N_{k+1}}(x_k)$ to $\nabla f_N(x_k)$. In fact, by [4, Theorem 6.2] $\nabla f_{N_{k+1}}(x_k)$ is sufficiently accurate with fixed probability at least $1-p_g$, i.e.,

$$Pr(\|\nabla f_N(x_k) - \nabla f_{N_{k+1}}(x_k)\| \le \chi_g) \ge 1 - p_g \quad \text{with} \quad \chi_g \in (0,1), \;\; p_g \in (0,1),$$

if the cardinality N_{k+1} satisfies

$$N_{k+1} \ge \min\left\{ N, \left\lceil \frac{2}{\chi_g} \left(\frac{V_g}{\chi_g} + \frac{2\zeta(x_k)}{3} \right) \log\left(\frac{n+1}{p_g} \right) \right\rceil \right\}, \tag{34}$$

with $E(\|\nabla \phi_i(x_k) - \nabla f_N(x_k)\|^2) \le V_g$ and $\max_{i \in \{1,\dots,N\}} |\nabla \phi_i(x)| \le \zeta(x)$, and $I_{N_{k+1}}$ is sampled uniformly in $\{1, 2, \dots, N\}$.

We conclude this section observing that, in the current form of the algorithm, at each iteration an attempt is made to use $N_{k+1} < N$ (see Step 2). By Corollary 2.5 we know that, for k sufficiently large, such a value will be rejected and this fact implies useless repetitions of Steps 2–5. To overcome this drawback, we replace (10) with

$$\begin{aligned} h(N_{k+1}) - h(\widetilde{N}_{k+1}) &\le \mu \Delta_k^{1+\gamma} && \text{if} \quad N_k \ne N \quad (35) \\ h(N_{k+1}) - h(N) &\le \min\{\mu \Delta_k^{1+\gamma}, \, \|\nabla f_N(x_k)\|\} && \text{if} \quad N_k = N \quad (36) \end{aligned}$$

Then, the following result holds.

Corollary 3.1 *Suppose (35) and (36) hold. For k sufficiently large, the use of sets $I_{N_{k+1}}$ of cardinality smaller than N is not attempted.*

Proof. By Corollary 2.5 and Corollary 2.6, we know that $N_k = N$ for all k sufficiently large and $\|\nabla f_N(x_k)\|$ tends to zero. Thus, letting k_* be the iteration index such that $\|\nabla f_N(x_k)\| < h(N-1)$, $\forall k \ge k_*$, it follows $N_{k+1} = N$, $\forall k \ge k_*$. □

4 Strongly convex problems

In this section we assume that f_N is strongly convex with strongly convex functions ϕ_i, $1 \leq i \leq N$, and analyze the local behaviour of IRETR method when full precision for the function and the gradient has been reached and a quadratic model of the following form is used:

$$m_k(p) = f_N(x_k) + \nabla f_N(x_k)^T p + \frac{1}{2} p^T \nabla^2 f_{D_{k+1}}(x_k) p,$$

with $1 \leq D_{k+1} \leq N$, $I_{D_{k+1}} \subseteq I_N$. Thus, we are focusing on the local behaviour of the trust-region method employing second order models with exact function and gradient and subsampled Hessian. Such a method has been investigated in [38] with respect to iteration complexity but not with respect to local convergence.

The additional assumptions used in this section are stated below.

Assumption 4.1 *The functions ϕ_i, $i = 1, \ldots, N$, are twice continuously differentiable and strongly convex in $\mathbb{R}^n$,*

$$\lambda_1 I \preceq \nabla^2 \phi_i(x) \preceq \lambda_n I, \quad \textit{with } 0 < \lambda_1 < \lambda_n, \tag{37}$$

where, given two matrices A and B, $A \preceq B$ means that $B - A$ is positive semidefinite.

Trivially, f_N is strongly convex and admits an unique minimizer x^*. Moreover, B_{k+1} is as in (31), both $\lambda_{\min}(B_{k+1}) \geq \lambda_1$ and $\lambda_{\max} \leq \lambda_n$ hold and Corollary 2.6 implies $\lim_{k\to\infty} x_k = x^*$.

The following theorem analyzes the behaviour of $\{x_k\}$ denoting

$$e(D_{k+1}) = \|\nabla^2 f_N(x_k) - \nabla^2 f_{D_{k+1}}(x_k)\|, \tag{38}$$

the error between $\nabla^2 f_N(x_k)$ and $\nabla^2 f_{D_{k+1}}(x_k)$. We also invoke the assumption below.

Assumption 4.2 *The Hessian $\nabla^2 f_N$ is Lipschitz continuous on $\mathcal{B}_\delta(x^*) := \{x \in \mathbb{R}^n : \|x - x^*\| \leq \delta\}$ with Lipschitz constant $2L_H$.*

Theorem 4.1 *Suppose that Assumptions 2.1, 2.2, 4.1, 4.2 hold. Let $\{x_k\}$ be generated by Algorithm* IRETR*, ε_g as in (8), β as in (22), η as in the Algorithm* IRETR *and B_{k+1} given by* (31).

i) Let $\epsilon \in (0,1)$ and D_{k+1} such that

$$\frac{1}{\tau \min\left\{\frac{\lambda_1^2}{4\beta}, \frac{\lambda_1}{2}\right\}} \left(2L_H\epsilon + e(D_{k+1})\right) \le 1-\eta. \tag{39}$$

Then, if k is sufficiently large, p_k is accepted in the first pass in Step 5 and $\mathcal{T}_k = 0$.

ii) There exist sufficiently small $\delta > 0$ and sufficiently large D such that, for all $x_k \in \mathcal{B}_\delta(x^*)$ and $D_{k+1} = D$, the error $\|x_k - x^*\|$ reduces linearly, i.e., $\|x_{k+1} - x^*\| < \tilde{\tau}\|x_k - x^*\|$ for some $\tilde{\tau} \in (0,1)$.

Proof. *i)* Let us consider k sufficiently large such that $N_{k+1} = N$ at termination of iteration k. Lemma 6.5.1 in [16] gives

$$\|p_k\| \le \frac{2}{\lambda_1}\|\nabla f_N(x_k)\|. \tag{40}$$

Let us consider the step p_k returned by iteration k. Combining (40) with (22) and (11) we obtain

$$m_k(0) - m_k(p_k) \ge \frac{1}{2}\omega\|p_k\|^2, \tag{41}$$

with $\omega = \tau \min\{\frac{\lambda_1}{2\beta}, 1\}\frac{\lambda_1}{2}$.

At Step 5 of the Algorithm, (14) has the form $f_N(x_k) - f_N(x_k + p_k) \ge \eta(m_k(0) - m_k(p_k))$. By Assumption 4.2 and (38), it follows

$$\begin{aligned}
\left|\frac{f_N(x_k) - f_N(x_k+p_k)}{m_k(0) - m_k(p_k)} - 1\right| &= \frac{|f_N(x_k+p_k) - m_k(p_k)|}{m_k(0) - m_k(p_k)} \\
&\le \frac{|\frac{1}{2}p_k^T(\nabla^2 f_N(x_k + tp_k) - \nabla^2 f_{D_{k+1}}(x_k))p_k|}{\frac{1}{2}\omega\|p_k\|^2} \\
&\le \frac{1}{2}\|p_k\|^2\Big(\frac{\|\nabla^2 f_N(x_k+tp_k) - \nabla^2 f_N(x_k)\|}{\frac{1}{2}\omega\|p_k\|^2} \\
&\quad + \frac{\|\nabla^2 f_N(x_k) - \nabla^2 f_{D_{k+1}}(x_k)\|}{\frac{1}{2}\omega\|p_k\|^2}\Big) \\
&\le \frac{2L_H\|p_k\| + e(D_{k+1})}{\omega},
\end{aligned}$$

where t is some scalar in $t \in (0,\,1)$ [16, Theorem 3.1.2]. Now, given $\epsilon \in (0,1)$ and D_{k+1} as in (39), (40) and Corollary 2.6 imply $\|p_k\| \le \epsilon$ for k large enough, say $k \geq \bar{k}$, and (39) implies the acceptance of the step. Then, Δ_k is not reduced and $\Delta_k \geq \Delta_{\bar{k}}$ for any $k \geq \bar{k}$.

ii) Using (40), Corollary 2.6 and item *i*) we can conclude that the trust-region bound becomes inactive for k sufficiently large, i.e., the step

$$p_k = -(\nabla^2 f_{D_{k+1}}(x_k))^{-1} \nabla f_N(x_k),$$

is accepted eventually. Consequently, using multivariate calculus results [18, Lemma 4.1.12] and Assumption 4.1

$$\begin{aligned}
\|x_{k+1} - x^*\| &= \|x_k - (\nabla^2 f_{D_{k+1}}(x_k))^{-1} \nabla f_N(x_k) - x^*\| \\
&= \|(\nabla^2 f_{D_{k+1}}(x_k))^{-1} (\nabla f_N(x^*) - \nabla f_N(x_k) - \nabla^2 f_{D_{k+1}}(x_k)(x^* - x_k)\| \\
&\leq \|(\nabla^2 f_{D_{k+1}}(x_k))^{-1}\| \left(\|\nabla f_N(x^*) - \nabla f_N(x_k) - \nabla^2 f_N(x_k)(x^* - x_k)\|\right. \\
&\quad \left. + \|(\nabla^2 f_N(x_k) - \nabla^2 f_{D_{k+1}}(x_k))(x^* - x_k)\|\right) \\
&\leq \frac{1}{\lambda_1} \|x_k - x^*\| \left(L_H \|x_k - x^*\| + e(D_{k+1})\right) \qquad (42)
\end{aligned}$$

Thus, the claim follows if δ and $D_{k+1} = D$ are such that $\tilde{\tau} := \frac{L_H \delta + e(D)}{\lambda_1} < 1$ and D satisfies (39).

□

Item *ii*) above may require a rather large value $D_{k+1} = D$ which is adverse for practical computation. A more stringent condition on D_{k+1} of the form $e(D_{k+1}) = O(\|\nabla f_N(x_k)\|)$ yields quadratic convergence but again such D_{k+1} might be very close to N. We next investigate on the more realistic situation where the Hessian accuracy requirement in (39) is guaranteed only with high-probability and provide a linear convergence result in expectation.

Let us now suppose that, given an accuracy requirement $\chi_H > 0$, the probability of $\|\nabla^2 f_N(x_k) - \nabla^2 f_{D_{k+1}}(x_k)\|$ being smaller than χ_H is larger than $1 - p_H$:

$$P(\|\nabla^2 f_N(x_k) - \nabla^2 f_{D_{k+1}}(x_k)\| \leq \chi_H) \geq 1 - p_H, \qquad (43)$$

for $p_H \in (0, 1)$. If the subsample $I_{D_{k+1}}$ is chosen randomly and uniformly, then the lower bound on the sample size ensuring (43) takes the form

$$D_{k+1} \geq \min \left\{ N, \left\lceil \frac{2}{\chi_H} \left(\frac{\lambda_n^2}{\chi_H} + \frac{\lambda_n}{3} \right) \log \left(\frac{2n}{p_H} \right) \right\rceil \right\}. \qquad (44)$$

The above bound is derived in [5, Lemma 3.1] and a similar bound is given in [3, Lemma 4].

We now provide a linear convergence result in expectation; the step p_k taken is the global minimizer of (6), i.e.,

$$(\nabla^2 f_{D_{k+1}}(x_k) + \nu_k I)p_k = -\nabla f_N(x_k),$$

for some $\nu_k \geq 0$, see [16, Theorem 7.2.1].

Theorem 4.2 *Suppose that Assumptions 2.1, 2.2, 4.1, 4.2 hold. Let $\{x_k\}$ be generated by Algorithm* IRETR *invoked with $\varepsilon_g = 0$ in (8), B_{k+1} as in* (31) *and p_k being the global minimizer of (6). If (43) holds and there exists a $\nu^* \in (0,1)$ such that for all k*

$$\frac{\nu_k}{\lambda_1 + \nu_k} \leq \nu^*, \tag{45}$$

then there exist δ, χ_H, p_H sufficiently small such that

$$E(\|x_{k+1} - x^*\|) \leq \bar{\tau} E(\|x_k - x^*\|), \tag{46}$$

for all k large enough and some $\bar{\tau} \in (0,1)$.

Proof. Take $\delta \in (0,1)$, $\chi_H > 0$, $p_H \in (0,1)$ such that

$$\rho = \frac{L_H\,\delta}{\lambda_1} + \frac{\chi_H}{\lambda_1} + \nu^* \leq \bar{\tau}, \tag{47}$$

$$p_H \leq \frac{(\bar{\tau} - \rho)}{1 + \frac{2\lambda_1}{\lambda_n}}. \tag{48}$$

for some $\bar{\tau} \in (0,1)$. Let k large enough such that $x_k \in \mathcal{B}_\delta(x^*)$.

Denote by A_k the event

$$\|\nabla^2 f_{D_{k+1}}(x_k) - \nabla^2 f_N(x_k)\| \leq \chi_H. \tag{49}$$

Then $P(A_k) \geq 1 - p_H$ and $P(\bar{A}_k) < p_H$, where $\bar{A}_k$ denotes the event A_k does not occur. If A_k happens then using multivariate calculus results [18, Lemma 4.1.12], Assumption 4.1, (45) and (47)

$$\begin{aligned}
\|x_{k+1} - x^*\| &= \|x_k - (\nabla^2 f_{D_{k+1}}(x_k) + \nu_k I)^{-1}\nabla f_N(x_k) - x^*\| \\
&= \|(\nabla^2 f_{D_{k+1}}(x_k) + \nu_k I)^{-1}(\nabla f_N(x^*) - \nabla f_N(x_k) - (\nabla^2 f_{D_{k+1}}(x_k) + \nu_k I)(x^* - x_k)\| \\
&\leq \|(\nabla^2 f_{D_{k+1}}(x_k) + \nu_k I)^{-1}\| \left(\|\nabla f_N(x^*) - \nabla f_N(x_k) - \nabla^2 f_N(x_k)(x^* - x_k)\| \right. \\
&\quad \left. + \|(\nabla^2 f_N(x_k) - \nabla^2 f_{D_{k+1}}(x_k))(x^* - x_k) + \nu_k(x^* - x_k)\|\right) \\
&\leq \frac{1}{\lambda_1 + \nu_k}\left(L_H\|x_k - x^*\| + e(D_{k+1}) + \nu_k\right)\|x_k - x^*\| \\
&\leq \left(\frac{L_H\,\delta}{\lambda_1} + \frac{\chi_H}{\lambda_1} + \nu^*\right)\|x_k - x^*\| \\
&= \rho\|x_k - x^*\|
\end{aligned} \tag{50}$$

Otherwise, if $\bar{A}_k$ is realized then by (40) we have

$$\|x_{k+1} - x^*\| \leq \left(1 + \frac{2\lambda_1}{\lambda_n}\right) \|x_k - x^*\|.$$

Therefore,

$$\begin{aligned} E(\|x_{k+1} - x^*\|) &= P(A_k)E(\|x_{k+1} - x^*\| | A_k) + P(\bar{A}_k)E(\|x_{k+1} - x^*\| | \bar{A}_k) \\ &\leq \rho E(\|x_k - x^*\|) + p_H \left(1 + \frac{2\lambda_1}{\lambda_n}\right) E(\|x_k - x^*\|) \\ &\leq \bar{\tau} E(\|x_k - x^*\|), \end{aligned}$$

where we have used (48) and $p(A_k) \leq 1$. □

5 Worst-case iteration and evaluation complexity to first-order critical points

In this section we provide an upper bound on the number of iterations and function-evaluations needed to find an ε_g-accurate first-order optimality point (8). The number of function-evaluations is intended as the number of evaluations of functions of the form f_M, for some $M \leq N$. We recall that a standard trust-region approach shows $\mathcal{O}(\varepsilon_g^{-2})$ worst-case iteration and full function complexity for first-order optimality [22].

Recalling that $h(N_k) - h(\widetilde{N}_{k+1}) = 0$ is equivalent to $N_k = \widetilde{N}_{k+1} = N$, consider the following partition of iteration indices k:

- $\mathcal{I}_1 = \{k \geq 0 \text{ s.t. } h(N_k) - h(\widetilde{N}_{k+1}) > 0\}$,
- $\mathcal{I}_2 = \{k \geq 0 \text{ s.t. } h(N_k) = h(\widetilde{N}_{k+1}) = 0, N_{k+1} = N \text{ and } \|\nabla f_N(x_k)\| > \varepsilon_g\}$,
- $\mathcal{I}_3 = \{k \geq 0 \text{ s.t. } h(N_k) = h(\widetilde{N}_{k+1}) = 0,\ N_{k+1} < N \text{ and } \|\nabla f_{N_{k+1}}(x_k)\| > \varepsilon_g\}$.

The value of N_{k+1} may change within iteration k before acceptance of the iterate; above N_{k+1} is the value at the end of iteration k, i.e., the value used for building the accepted iterate x_{k+1}.

Our analysis is carried out fixing $\gamma = 1$ in Algorithm IRETR and the first result provides a lower bound on the trust-region radius at termination of iteration k.

Lemma 5.1 *Let Assumptions 2.1–2.4 hold. Suppose furthermore $\gamma = 1$ in Algorithm* IRETR. *Then,*

i) for any $k \in \mathcal{I}_1$

$$\Delta_k \geq \min\left\{\zeta_1\sqrt{\frac{\eta(1-\eta)}{\kappa_T+\mu}(1-r)\underline{h}},\, \Delta_0\right\},$$

ii) for any $k \in \mathcal{I}_2 \cup \mathcal{I}_3$,

$$\Delta_k \geq \min\left\{\zeta_1\sqrt{\frac{\underline{h}}{\mu}},\, \zeta_1\Gamma\varepsilon_g,\, \Delta_0\right\}, \tag{51}$$

for some positive Γ and μ as in the Algorithm.

Proof. The initial Δ_k may be reduced in Steps 3 and 5 of the Algorithm. Step 3 is performed only if $k \in \mathcal{I}_3$.

Let us consider case *i*). Since $\gamma = 1$ equation (24) becomes

$$|\theta_{k+1}(m_k(p_k)-f_{N_{k+1}}(x_k+p_k))+(1-\theta_{k+1})(h(\widetilde{N}_{k+1})-h(N_{k+1}))| \leq (\kappa_T+\mu)\Delta_k^2.$$

From (23), inequality (14) is satisfied whenever

$$\Delta_k \leq \sqrt{\frac{\eta(1-\eta)}{\kappa_T+\mu}(h(N_k)-h(\widetilde{N}_{k+1}))}\,.$$

Thus, using (9), if

$$\Delta_k \leq \sqrt{\frac{\eta(1-\eta)}{\kappa_T+\mu}(1-r)\underline{h}},$$

then (14) holds and the claim *i*) follows from the rule for decreasing Δ_k in Step 5 of Algorithm IRETR.

Let us consider case *ii*). Concerning Step 3, it is performed as long as $N_{k+1} < N$. Then, (10) ensures that at termination of the loop in Steps 2–3

$$\Delta_k \geq \zeta_1\sqrt{\frac{\underline{h}}{\mu}}\ .$$

Concerning Step 5, first suppose $k \in \mathcal{I}_2$ and $\Delta_k \leq \varepsilon_g/\beta$ with β as in (22). Using (25) we can conclude that if

$$\Delta_k \leq \frac{\tau\underline{\theta}(1-\eta)}{2\kappa_T}\,\varepsilon_g,$$

then (14) is satisfied.

Suppose now $k \in \mathcal{I}_3$ and $\Delta_k \le \varepsilon_g/\beta$. Using $\gamma = 1$, equation (27) becomes

$$\mathrm{Ared}_k(\theta_{k+1}) - \eta \mathrm{Pred}_k(\theta_{k+1}) \ge \left(\frac{1}{2}\tau\underline{\theta}(1-\eta)\|\nabla f_{N_{k+1}}(x_k)\| - (\kappa_T + \mu)\Delta_k \right) \Delta_k,$$

and if

$$\Delta_k \le \frac{\tau\underline{\theta}(1-\eta)}{2(\kappa_T + \mu)}\, \varepsilon_g,$$

then (14) is satisfied.

The upper bound on Δ_k for $k \in \mathcal{I}_3$ is sharper than the one obtained for $k \in \mathcal{I}_2$. Then, due to the rule used to decrease Δ_k in Step 5, we can conclude that, at iteration $k \in \mathcal{I}_2 \cup \mathcal{I}_3$, condition (14) is satisfied if

$$\Delta_k > \zeta_1 \min\left\{ \frac{1}{\beta}, \frac{\tau\underline{\theta}(1-\eta)}{2(\kappa_T + \mu)} \right\} \varepsilon_g \overset{\text{def}}{=} \zeta_1 \Gamma \varepsilon_g, \tag{52}$$

and the claim follows. □

Theorem 5.2 *Let Assumptions 2.1–2.4 hold. Suppose furthermore $\gamma = 1$ in Algorithm* IRETR *and let f_{low} the lower bound of f_N in Ω. Then,*

i) the cardinality $|\mathcal{I}_1|$ satisfies

$$|\mathcal{I}_1| \le \left\lceil \nu_1 \underline{h}^{-1} \right\rceil,$$

with $\nu_1 = \frac{\xi - \theta f_{low}}{\eta^2(1-r)}$, ξ as in (30), $\underline{\theta}$ as in Lemma 2.2, η and r as in the Algorithm IRETR.

ii) the cardinality $|\mathcal{I}_2| + |\mathcal{I}_3|$ satisfies

$$|\mathcal{I}_2| + |\mathcal{I}_3| \le \begin{cases} \left\lceil \nu_2\, \varepsilon_g^{-2} \right\rceil & \text{if } \Gamma\varepsilon_g \le \min\left\{ \sqrt{\dfrac{\underline{h}}{\mu}}, \dfrac{\Delta_0}{\zeta_1} \right\}, \\ \nu_3 \underline{h}^{-\frac{1}{2}} \varepsilon_g^{-1} & \text{if } \sqrt{\dfrac{\underline{h}}{\mu}} \le \min\left\{ \Gamma\varepsilon_g, \dfrac{\Delta_0}{\zeta_1}, \right\} \end{cases}$$

with positive $\nu_2 = \frac{2}{\eta\Gamma}\left(f_{N_0}(x_0) - f_{low} + (\sigma\eta + 1 - \underline{\theta})\frac{\xi - \theta f_{low}}{\eta^2(1-r)} \right)$, $\nu_3 = \nu_2 \Gamma \sqrt{\mu}$.

Proof. Let us denote with $\bar{k}$ the last iterate of Algorithm IRETR and note that $N_{\bar{k}} = N$ by definition of the algorithm. From (29) it follows

$$\sum_{k=0}^{\bar{k}-1} \text{Pred}_k(\theta_{k+1}) \le \frac{\xi - \theta_{\bar{k}+1} f_N(x_{\bar{k}})}{\eta} \le \frac{\xi - \underline{\theta} f_{low}}{\eta}, \quad \forall k \ge 0,$$

and consequently (28) yields

$$\sum_{k=0}^{\bar{k}-1} h(N_k) \le \frac{\xi - \underline{\theta} f_{low}}{\eta^2(1-r)}. \tag{53}$$

Then the number of indices k such that $h(N_k) > \underline{h}$ is bounded above by

$$\frac{\xi - \underline{\theta} f_{low}}{\underline{h}\eta^2(1-r)},$$

and $i)$ follows.

Let us consider the case $k \in \mathcal{I}_2 \cup \mathcal{I}_3$. Note that by (16), (14), (15), (19) and (11), we have

$$\begin{array}{rcl} Ared_k(\theta_{k+1}) & = & \theta_{k+1}(f_N(x_k) - f_{N_{k+1}}(x_{k+1})) - (1-\theta_{k+1})h(N_{k+1}) \\ & \ge & \eta\theta_{k+1}(f_N(x_k) - f_{N_{k+1}}(x_k) + m_k(0) - m_k(p_k)) \\ & \ge & -\sigma\eta\theta_{k+1}h(N_{k+1}) + \eta\theta_{k+1}(m_k(0) - m_k(p_k)) \\ & \ge & -\sigma\eta\theta_{k+1}h(N_{k+1}) + \tau\eta\theta_{k+1}(m_k(0) - m_k(p_k^C)) \end{array}$$

Then, by using (51) and (22) it follows

$$f_N(x_k) - f_{N_{k+1}}(x_{k+1}) + \sigma\eta h(N_{k+1}) \ge \frac{\tau\eta}{2} \min\left\{\zeta_1\Gamma\varepsilon_g, \zeta_1\sqrt{\frac{\underline{h}}{\mu}}, \Delta_0\right\}\varepsilon_g. \tag{54}$$

Moreover, note that due to the definition of $Ared_k(\theta_{k+1})$ and inequalities (17) and (14), the following inequality holds at termination of each iteration $k \ge 0$:

$$\frac{Ared_k(\theta_{k+1})}{\theta_{k+1}} = f_{N_k}(x_k) - f_{N_{k+1}}(x_{k+1}) + \frac{1-\theta_{k+1}}{\theta_{k+1}}(h(N_k) - h(N_{k+1})) \ge 0 \tag{55}$$

Then, since $\frac{Ared_k(\theta_{k+1})}{\theta_{k+1}}$ is positive,

$$\sum_{k \in \mathcal{I}_2 \cup \mathcal{I}_3} \frac{Ared_k(\theta_{k+1})}{\theta_{k+1}} \le \sum_{k=0}^{\bar{k}-1} \frac{Ared_k(\theta_{k+1})}{\theta_{k+1}},$$

and this implies

$$\begin{aligned}
\sum_{k\in\mathcal{I}_2\cup\mathcal{I}_3} \big(f_N(x_k) - f_{N_{k+1}}(x_{k+1})\big) &\le \sum_{k=0}^{\bar{k}-1} \big(f_{N_k}(x_k) - f_{N_{k+1}}(x_{k+1})\big) \\
+\sum_{k=0}^{\bar{k}-1} \frac{1-\theta_{k+1}}{\theta_{k+1}}(h(N_k) - h(N_{k+1})) &- \sum_{k\in\mathcal{I}_2\cup\mathcal{I}_3} \frac{1-\theta_{k+1}}{\theta_{k+1}}(h(N_k) - h(N_{k+1})) \\
= \sum_{k=0}^{\bar{k}-1} \big(f_{N_k}(x_k) - f_{N_{k+1}}(x_{k+1})\big) &+ \sum_{k\in\mathcal{I}_1} \frac{1-\theta_{k+1}}{\theta_{k+1}}(h(N_k) - h(N_{k+1})) \\
\le \sum_{k=0}^{\bar{k}-1} \big(f_{N_k}(x_k) - f_{N_{k+1}}(x_{k+1})\big) &+ \frac{1-\underline{\theta}}{\underline{\theta}} \sum_{k=0}^{\bar{k}-1} h(N_k).
\end{aligned}$$

This implies

$$\begin{aligned}
&\sum_{k=0}^{\bar{k}-1} \big(f_{N_k}(x_k) - f_{N_{k+1}}(x_{k+1})\big) + \frac{1-\underline{\theta}}{\underline{\theta}} \sum_{k=0}^{\bar{k}-1} h(N_k) \ge \\
&\sum_{k\in\mathcal{I}_2\cup\mathcal{I}_3} \big(f_N(x_k) - f_{N_{k+1}}(x_{k+1})\big) \qquad (56)
\end{aligned}$$

Then, (56), (53), (54) and $h(N_{\bar{k}}) = 0$ yield

$$\begin{aligned}
&f_{N_0}(x_0) - f_{low} + \left(\sigma\eta + \frac{1-\underline{\theta}}{\underline{\theta}}\right) \frac{\xi - \underline{\theta} f_{low}}{\eta^2(1-r)} \\
&\ge \sum_{k=0}^{\bar{k}-1} \big(f_{N_k}(x_k) - f_{N_{k+1}}(x_{k+1})\big) + \left(\sigma\eta + \frac{1-\underline{\theta}}{\underline{\theta}}\right) \sum_{k=0}^{\bar{k}-1} h(N_k) \\
&\ge \sum_{k\in\mathcal{I}_2\cup\mathcal{I}_3} \big(f_N(x_k) - f_{N_{k+1}}(x_{k+1}) + \sigma\eta h(N_{k+1})\big) \\
&\ge (|\mathcal{I}_2| + |\mathcal{I}_3|)\frac{\eta}{2} \min\left\{\zeta_1\Gamma\varepsilon_g, \zeta_1\sqrt{\frac{\underline{h}}{\mu}}, \Delta_0\right\}\varepsilon_g,
\end{aligned}$$

and claim *ii*) follows. □

Considering that ε_g is an optimality measure and $\underline{h}$ is expected to be small, it is reasonable to suppose that

$$\Delta_0 \ge \zeta_1 \max\left\{\Gamma\varepsilon_g, \sqrt{\frac{\underline{h}}{\mu}}\right\}. \qquad (57)$$

Under this condition, Theorem 5.2 gives the iteration complexity

$$|\mathcal{I}_1| + |\mathcal{I}_2| + |\mathcal{I}_3| = \mathcal{O}\left(\underline{h}^{-1} + \max\{\varepsilon_g^{-2},\, \underline{h}^{-\frac{1}{2}}\varepsilon_g^{-1}\}\right).$$

As a consequence, for suitable values of $\underline{h}$, the worst-case iteration complexity $\mathcal{O}(\varepsilon_g^{-2})$ of the standard trust-region method is retained, despite inaccuracy in functions and gradients. This result is stated below, where we count the number of iterations needed to satisfy $\|\nabla f_N(x_k)\| \leq \varepsilon_g$ or $\|\nabla f_{N_{k+1}}(x_k)\| \leq \varepsilon_g$ and $N_k = N$, i.e., iterations in $\mathcal{I}_1 \cup \mathcal{I}_2 \cup \mathcal{I}_3$ and iteration $\bar{k}$.

Corollary 5.3 *Let Assumptions 2.1–2.4 hold. Assume furthermore $\gamma = 1$ in Algorithm* IRETR. *Then, there exists a constant $\nu_4 > 0$ such that Algorithm* IRETR *needs at most*

$$\lceil \nu_4 \varepsilon_g^{-2} \rceil + 1$$

iterations, provided that $\underline{h}^{-1} = \mathcal{O}(\varepsilon_g^{-2})$ and (57) holds.

In case $h(M) = (N-M)/N$, it holds $\underline{h} = 1/N$ and $\underline{h}^{-1} = \mathcal{O}(\varepsilon_g^{-2})$ implies $N = \mathcal{O}(\varepsilon_g^{-2})$. In case N is larger, the number of iterations taken before full-accuracy is reached may deteriorate the complexity of the standard trust-region approach.

In order to derive the worst-case function evaluation complexity we need to bound the total number of trust-region reductions as each trust-region reduction calls for one (possibly subsampled) function evaluation at trial point $x_k + p_k$.

Theorem 5.4 *Let Assumptions 2.1–2.4 hold. Assume furthermore $\gamma = 1$ in Algorithm* IRETR *and let $\mathcal{T}_j$ be the number of trust-region reductions at a generic iteration j of the algorithm. Then, for any $k \geq 1$,*

$$\sum_{j=0}^{k} \mathcal{T}_j \leq \left\lceil \frac{\log(\underline{\Delta}/\Delta_0)}{\log(\zeta_1)} - k\frac{\log(\zeta_2)}{\log(\zeta_1)} \right\rceil,$$

where

$$\underline{\Delta} = \min\left\{\zeta_1\sqrt{\frac{\eta(1-\eta)}{\kappa_T + \mu}(1-r)\underline{h}}, \zeta_1\sqrt{\frac{\underline{h}}{\mu}}, \zeta_1\Gamma\varepsilon_g, \Delta_0\right\}.$$

Proof. Let us proceed by induction. By the updating rules of the trust-region radius in Step 5 of Algorithm IRETR, at termination of the iteration $j = 0$ we have

$$\Delta_1 \in [\zeta_1^{\mathcal{T}_0}\Delta_0, \zeta_2\zeta_1^{\mathcal{T}_0}\Delta_0].$$

Then, assume that at iteration $k \geq 1$

$$\Delta_k \in [\zeta_1^{w_k}\Delta_0, \zeta_2^k \zeta_1^{w_k}\Delta_0], \tag{58}$$

with $w_k = \sum_{j=0}^{k-1} \mathcal{T}_j$. At the end of iteration k, after $\mathcal{T}_k$ reductions of the trust-region radius we have

$$\Delta_{k+1} \in [\zeta_1^{\mathcal{T}_k}\Delta_k, \zeta_2\zeta_1^{\mathcal{T}_k}\Delta_k],$$

and consequently,

$$\Delta_{k+1} \in [\zeta_1^{w_{k+1}}\Delta_0, \zeta_2^{k+1}\zeta_1^{w_{k+1}}\Delta_0],$$

i.e., (58) holds for any $k \geq 1$. Taking into account that Lemma 5.1 ensures that iteration k terminates with $\Delta_k \geq \underline{\Delta}$, in the adverse case where the initial Δ_k is given by $\zeta_2^k\zeta_1^{w_k}\Delta_0$ (see (58)), at termination of iteration k we are ensured that

$$\zeta_2^k\zeta_1^{w_{k+1}}\Delta_0 \geq \underline{\Delta}.$$

This yields the thesis, taking into account that $\zeta_1 < 1$. □

Using the previous results we can now state our function evaluation complexity result.

Corollary 5.5 *Let Assumptions 2.1–2.4 hold. Assume furthermore $\gamma = 1$ in Algorithm* IRETR. *Then, if $\underline{h}^{-1} = \mathcal{O}(\varepsilon_g^{-2})$ and Δ_0 satisfies* (57) *and it is independently of ε_g, there exists a constant ν_5 such that Algorithm* IRETR *needs at most*

$$\left\lceil \nu_4\varepsilon_g^{-2}\left(1 - \frac{\log(\zeta_2)}{\log(\zeta_1)}\right) - \frac{\log(\nu_5\varepsilon_g^{-1})}{\log(\zeta_1)} \right\rceil$$

function evaluations, where ν_4 is given in Corollary 5.3.

Proof. Assumption $\underline{h}^{-1} = \mathcal{O}(\varepsilon_g^{-2})$, (57) and Δ_0 independent of ε_g ensure $\underline{\Delta} = \nu_5\varepsilon_g$, for some positive ν_5. Then Corollary 5.3 and Theorem 5.4 yield the thesis. □

6 Numerical experiments

In this section we report on our numerical experience with Algorithm IRETR employing the second order model (5) and D_{k+1} equal to a fixed fraction of N_{k+1}. Our aim is to show that our adaptive and deterministic strategy for choosing the sample size N_k and the use of subsampled functions, gradients and Hessians is effective and provides a gain in the overall computational cost with respect to a standard trust-region approach. To this end, we compare our method with "standard" trust-region implementations, i.e. implementations where functions and gradients are computed at full accuracy too. Specifically, we compare with the implementation, named STATR_SH, employing full functions and gradients and subsampled Hessian B_k as in (31) with $D_{k+1} = \lceil 0.1N \rceil$, and with the implementation, named STATR_FH, where functions, first and second order derivatives are computed at full accuracy.

All the results have been obtained running a Matlab R2019b code on an Intel Core i5-6600K CPU 3.50 GHz x 4, 16.0GB RAM.

6.1 Test problems

We tested our method both on convex and nonconvex problems arising in binary classification problems. Let $\{(a_i, b_i)\}_{i=1}^N$ denote the pairs forming the data set with $a_i \in \mathbb{R}^n$ being the vector containing the entries of the i-th example and b_i being its label. The data set we employed are displayed in Table 1. In the table for each data set we report the number N of training examples and the dimension n of each instance. Moreover we report the number of elements in the testing set N_T.

We performed a logistic regression to solve classification problems associated to the data sets MUSHROOMS, CINA0 and GISETTE. In this case $b_i \in \{-1, +1\}$ and the strongly convex objective function is given by the logistic loss with ℓ_2-regularization

$$f_N(x) = \frac{1}{N} \sum_{i=1}^{N} \log(1 + e^{-b_i a_i^T x}) + \frac{1}{2N} \|x\|^2.$$

Classification problems associated with the remaining data sets were solved using the sigmoid function and least-squares loss. Here $b_i \in \{0, +1\}$ and the non-convex objective function has the form

$$f_N(x) = \frac{1}{N} \sum_{i=1}^{N} \left(b_i - \frac{1}{1 + e^{-a_i^T x}} \right)^2.$$

6.2 Implementation issues

The trust-region parameters of the procedures under comparison are fixed as

$$\Delta_0 = 10, \quad \tau = 0.1, \quad \eta = 0.1, \quad \zeta_1 = 0.5, \quad \zeta_2 = 1.2.$$

	Training set		Testing set
Data set	N	n	N_T
Mushrooms [24]	5000	112	3124
Cina0 [14]	10000	132	6033
Gisette [24]	5000	5000	1000
A9a [24]	22793	123	9768
Covertype [24]	464810	54	116202
Ijcnn1 [15]	49990	22	91701
Mnist [23]	60000	784	10000
Htru2 [24]	10000	8	7898

Table 1: Data sets used

The trust-region problem is solved approximately using CG-Steihaug method [16]. The Conjugate Gradient (CG) method is applied without preconditioning and the procedure is stopped when the relative residual becomes smaller than 10^{-3} or a maximum of 100 iterations is performed. In Step 5, in case of successful iterations, we update the trust-region radius as follows. If $\text{Ared}_k(\theta_{k+1})/\text{Pred}_k(\theta_{k+1}) \geq 1.1$ we set $\Delta_{k+1}^{(0)} = \zeta_2 \Delta_k^{(\mathcal{T}_k)}$, otherwise we set $\Delta_{k+1}^{(0)} = \Delta_k^{(\mathcal{T}_k)}$.

Focusing on Algorithms IRETR, we tested two rules for choosing the sample size. In the first implementation, later referred to as IRETR_D, the sample size varies dynamically. The infeasibility measure h and the initialization parameters for inexact restoration are:

$$h(M) = \frac{N - M}{N}, \quad N_0 = \lceil 0.1\, N \rceil, \quad \theta_0 = 0.9.$$

The parameters $\gamma = 1$, $\mu = 100/N$ are used in (10). The updating rules for choosing $\widetilde{N}_{k+1}$, N_{k+1} in Steps 1 and 2 are the following:

$$\begin{aligned}
\widetilde{N}_{k+1} &= \min\{N, \lceil 1.2\, N_k \rceil\}, \\
N_{k+1} &= \begin{cases} \left\lceil \widetilde{N}_{k+1} - 10^2 \Delta_k^{1+\gamma} \right\rceil & \text{if } \left\lceil \widetilde{N}_{k+1} - 10^2 \Delta_k^{1+\gamma} \right\rceil \in [N_0, 0.95N], \\ \widetilde{N}_{k+1} & \text{if } \left\lceil \widetilde{N}_{k+1} - 10^2 \Delta_k^{1+\gamma} \right\rceil < N_0, \\ N & \text{if } \left\lceil \widetilde{N}_{k+1} - 10^2 \Delta_k^{1+\gamma} \right\rceil > 0.95N. \end{cases}
\end{aligned}$$

We note that the choice of $\widetilde{N}_{k+1}$ falls into (9) with $r = (N - 0.2)/N$.

In the second implementation, we set again

$$h(M) = \frac{N - M}{N}, \quad \theta_0 = 0.9.$$

Then, the sample size N_{k+1} is increased according the geometric growth:

$$N_0 = \lceil 0.1\, N \rceil\,, \qquad N_{k+1} = \widetilde{N}_{k+1} = \min\{N, \lceil 1.2\, N_k \rceil\}.$$

We will refer to this implementation as IRETR_GG. We note that this choice of N_{k+1} amount to choosing $\mu = 0$ in (10).

In both implementations IRETR_D and IRETR_GG the first time that $N_k = N_{k+1} = N$ occurs, then the value of the trust-region radius is set to $\Delta_k^{(\mathfrak{T}_k)} = \max\{1,\, \Delta_k^{(\mathfrak{T}_k)}\}$. Moreover, the Hessian matrix B_k is formed via (31) with

$$D_{k+1} = \lceil 0.1\, N_{k+1} \rceil\,, \ \ \forall k \geq 0.$$

Thus, the Hessian sample size changes dynamically until the full sample for function and gradient is reached. The sets $I_{N_{k+1}}$ and $I_{D_{k+1}}$ are generated using the `Matlab` function `randsample` with no replacement. When the sample size N_{k+1} is increased, the new sample set can be computed from scratch or can be obtained randomly adding new samples to the previous sample set. Despite this latter choice produces computational savings, in view of a truly random process we generate each $I_{N_{k+1}}$ from scratch.

Concerning the stopping criteria, for all the algorithms under comparison, we imposed a maximum of 1000 iterations and we declared a successful termination when one of the two following conditions is met

$$\|\nabla f_{N_k}(x_k)\| \leq \varphi, \qquad |f_{N_k}(x_k) - f_{N_{k-1}}(x_{k-1})| \leq \varphi |f_{N_k}(x_k)|, \tag{59}$$

with $\varphi = 10^{-4}$. We underline that for IRETR_D and IRETR_GG the above checks are on possibly subsampled functions and gradients and we allow for termination before full precision is reached.

The initial guess is $x_0 = (0, \ldots, 0)^T$ for all runs.

6.3 Numerical results

The first set of results presented shows the performance of Algorithms IRETR_D, IRETR_GG, STATR_SH and STATR_FH. In our test problems, the main cost in the computation of ϕ_i for any $1 \leq i \leq N$ is the scalar product $a_i^T x$. Once this product is evaluated, it can be reused for computing $\nabla \phi_i$ and $\nabla^2 \phi_i$. In particular, computing $\nabla^2 \phi_i$ times a vector v at each CG iteration requires a scalar product $a_i^T v$ i.e., it is as expensive as evaluating ϕ_i.

Data set	`nfe`	`nfe(save)`		
	iretr_d	iretr_gg	statr_sh	statr_fh
Mushrooms	27	30 (10%)	51 (47%)	108 (75%)
Cina0	88	99 (11%)	96 (8%)	416 (78%)
Gisette	346	362 (4%)	432 (20%)	594 (42%)
A9a	22	25 (12%)	45 (51%)	445 (95%)
Covertype	17	23 (26%)	48 (65%)	698 (98%)
Ijcnn1	20	25 (20%)	36 (44%)	128 (84%)
Mnist	46	50 (8%)	58 (20%)	955 (95%)
Htru2	38	37 (-3%)	43 (12%)	87 (56%)

Table 2: Function evaluations performed by iretr_d, iretr_gg, statr_sh and statr_fh and saving obtained by iretr_d over iretr_gg, statr_sh and statr_fh.

Therefore, if one full function evaluation is denoted as `nfe`, computing f_M costs $\frac{M}{N}\texttt{nfe}$ while each CG iteration costs $\frac{D_{k+1}}{N}\texttt{nfe}$. Since the selection of sets $I_{N_{k+1}}$ and $I_{D_{k+1}}$ in Algorithms iretr_d, iretr_gg and statr_sh is random, the cost associated to such algorithms is measured on average over 50 runs.

In Table 2 for each method and for each data set we report the number `nfe` of full function evaluations performed and the percentage of saving obtained by Algorithm iretr_d with respect to iretr_gg, statr_sh and to statr_fh. First, we can observe that Algorithm iretr_d is in general less costly than the variant iretr_gg; this indicates that the dynamic choice of the sample size, aiming to make slow progress to full precision, is effective and does not deteriorate the performance of iretr when the geometrical growth of the sample size is the most effective (see the results for Htru2). Second, we observe a remarkable saving of both iretr_d and iretr_gg with respect to the full standard trust-region for all the data sets used; compared to statr_sh the saving is lower, as expected, but still considerable overall.

To give more insight into the two implementations iretr_d, in Figures 1 and 2 we plot the sample size N_k versus the iterations for Mushrooms and A9a problems. The dashed line plots $N_{k+1} = \lceil (1.2)^k N_0 \rceil$ versus iterations, that is the sample size corresponding to the geometric growth used in iretr_gg. The increase of N_k along iterations in iretr_d is considerably slower than that provided by the geometric growth; in two runs, the cardinality N_k in iretr_d reaches the value N, as expected from the theory,

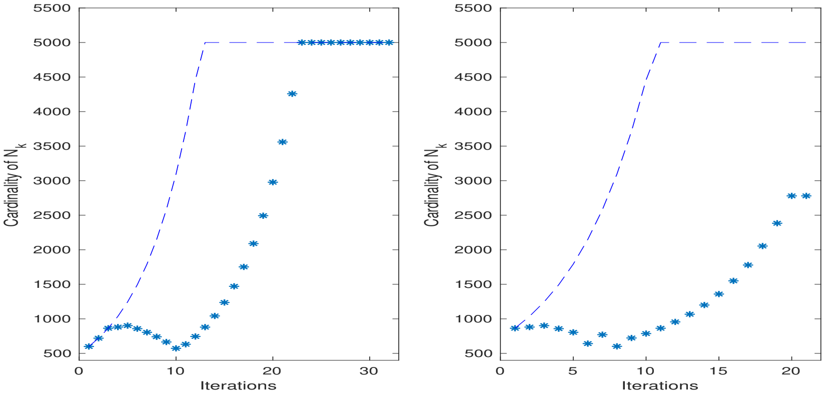

Figure 1: Mushrooms data set. N_k versus iretr_d iterations (" * "), sample size $N_{k+1} = (1.2)^k N_0$ (dashed line).

but in the first phase of the iterative process it is a small fraction of N and decreases at some iterations. In the other two runs, iretr_d does not reach full precision, iterations terminate with a cardinality $N_{k+1} = 2780$, corresponding to the 56% of the training set and $N_{k+1} = 16495$, corresponding to the 72% of the training set, respectively. In fact, despite the adaptive strategy of iretr yields $N_k = N$ for k sufficiently large, our stopping rule (59) is applied on possibly subsampled functions and gradients. This feature is in accordance with the motivations for using subsampling: data in a training set show redundancy and in general using subsets of the sample data is enough to provide a small testing error. At this regard, consider Figure 3 related to the data set Mushrooms, $N = 5000$. At each iteration and for three runs corresponding to different sample sizes at termination, we plot the training loss $f_{N_k}(x_k)$ versus the value of N_k; at termination: N_k =1941 (dashed line), N_k= 4241 (dash-dotted line), $N_k = N$ (solid line). We also display the testing loss f_{N_T} at termination. Although in two runs the final sample size is approximately 39% and 85% of the data in the training set, interestingly the testing loss is in between $1 \cdot 10^{-1}$ and $3 \cdot 10^{-1}$ in all runs. Thus, monitoring the values of subsampled functions and gradients in (59) is effective.

The previous discussion is supported by further observations. In Figure 4, we plot the value of the training loss versus the number of function evaluations required to solve Mushrooms and Htru2 problems with iretr_d,

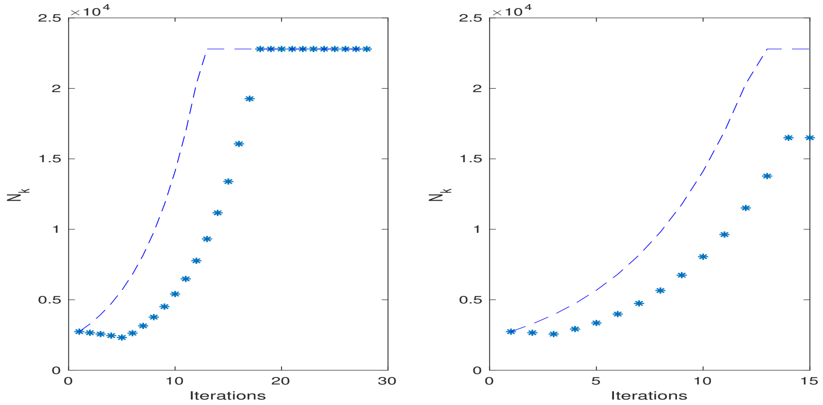

Figure 2: A9A data set. N_k versus IRETR_D iterations (“ * ”), sample size $N_{k+1} = (1.2)^k N_0$ (dashed line).

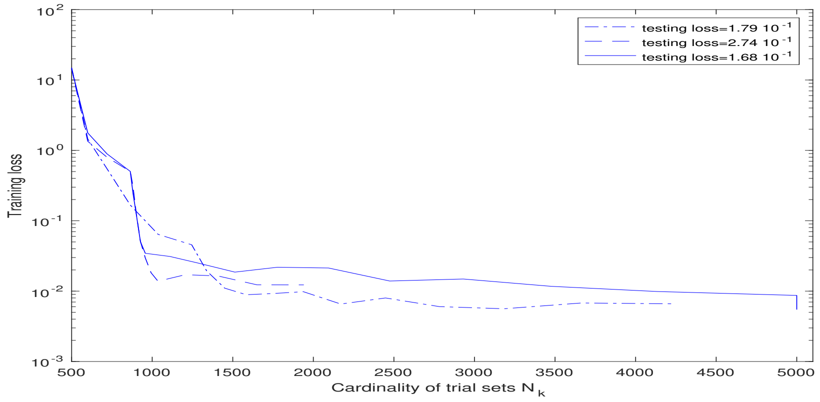

Figure 3: MUSHROOMS data set, N=5000. Training loss versus N_k and testing loss at termination using IRETR_D. Values of N_k at termination: 1941 (dashed line); 4241 (dash-dotted line); 5000 (solid line).

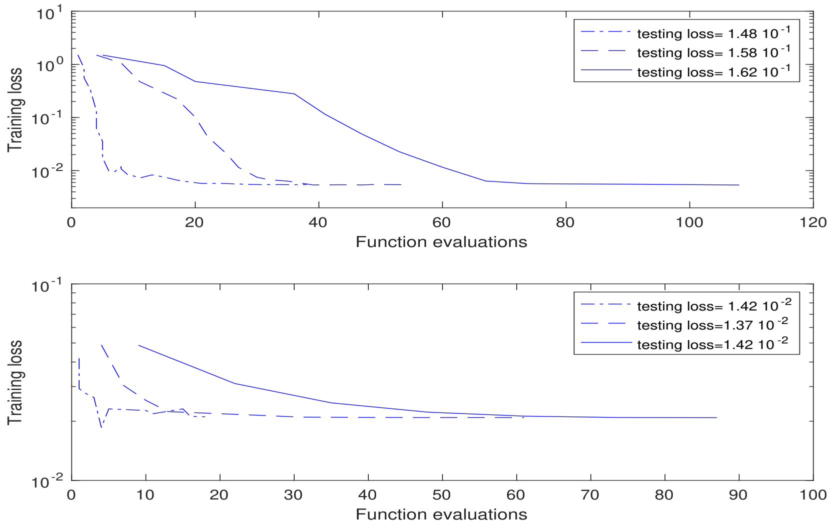

Figure 4: Training loss versus function evaluations and testing loss: iretr_d (dash-dotted line); statr_sh (dashed line) and statr_fh (solid line). Upper: Mushrooms data set, lower: Htru2 data set.

statr_sh and statr_fh. In these runs, iretr_d terminates with $N_k = N$ in Mushrooms problem while terminates with $N_k = 7426$ (74% of the samples) in Htru2 problem. At termination, the values of both the training loss and the testing loss provided by the three methods are similar and this feature further supports both termination before full precision is reached and the inexact restoration approach for handling subsampled functions and derivatives.

Finally, Figure 5 refers to the dataset Cina0 and displays the values of the training and testing logistic loss along the iterations of iretr_d using the tolerance $\varphi = 10^{-8}$ in (59). In the progress of the iterations the loss values settle and performing the last thirteen iterations is pointless.

Acknowledgement Dedicated with friendship to José Mario Martínez for his outstanding scientific contributions.

References

[1] Bastin F., Cirillo C., Toint P.L., An adaptive Monte Carlo algorithm for computing mixed logit estimators, Computational Management Science

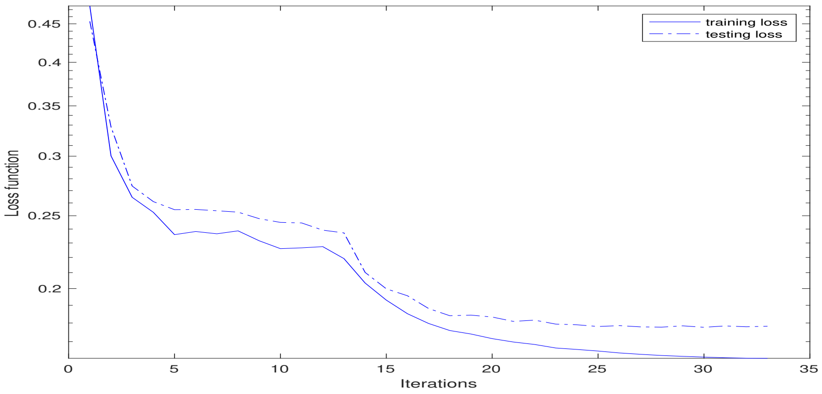

Figure 5: Cina0 data set: training and testing loss function versus iterations computed by itetr_d. Stopping threshold $\varphi = 10^{-8}$.

3(1), 55-79, 2006.

[2] Bastin F., Cirillo C., Toint P.L., Convergence theory for nonconvex stochastic programming with an application to mixed logit, Mathematical Programming, 108, 207-234, 2006.

[3] Bellavia, S., Gurioli, G., Morini, B., Adaptive cubic regularization methods with dynamic inexact Hessian information and applications to finite-sum minimization, IMA J. Numerical Analysis, 2020, drz076, https://doi.org/10.1093/imanum/drz076

[4] Bellavia, S., Gurioli, G., Morini, B., Toint, Ph.L., Adaptive regularization algorithms with inexact evaluations for nonconvex optimization, SIAM Journal on Optimization, 29(4), pp. 2281–2915, 2019.

[5] Bellavia, S., Krejić, N., Krklec Jerinkić, N., Subsampled Inexact Newton methods for minimizing large sums of convex function, IMA Journal of Numerical Analysis, 2019, https://doi.org/10.1093/imanum/drz027

[6] Berahas A. S., Bollapragada R., Nocedal J., An Investigation of Newton-Sketch and Subsampled Newton Methods, Optimization Methods and Software, 2020, https://doi.org/10.1080/10556788.2020.1725751

[7] Birgin, G.E., Krejić, N., Martínez, J.M., On the employment of Inexact Restoration for the minimization of functions whose evaluation is

subject to programming errors, Mathematics of Computation 87(311), 1307-1326, 2018.

[8] Birgin, G.E., Krejić, N., Martínez, J.M., Iteration and evaluation complexity on the minimization of functions whose computation is intrinsically inexact, Mathematics of Computation, 89, 253-278, 2020.

[9] Blanchet J., Cartis C., Menickelly M., Scheinberg K., Convergence rate analysis of a stochastic trust region method via supermartingales, Informs Journal on Optimization, 1(2), 92–119, 2019.

[10] Bollapragada, R., Byrd, R., Nocedal, J., Exact and Inexact Subsampled Newton Methods for Optimization, IMA Journal of Numerical Analysis, 39(20), 545-578, 2019.

[11] Bottou, L., Curtis F.C., Nocedal, J. Optimization Methods for Large-Scale Machine Learning, SIAM Review, 60(2), 223-311, 2018.

[12] Byrd R.H., Hansen S.L., Nocedal J., Singer Y., A Stochastic Quasi-Newton Method for Large-Scale Optimization, SIAM Journal on Optimization, 26(2), 1008-1021, 2016.

[13] Byrd R.H., Chin G.M., Nocedal J., Wu Y., Sample size selection in optimization methods for machine learning, Mathematical Programming, 134(1), 127-155, 2012.

[14] Causality workbench team, A marketing dataset, `http://www.causality.inf.ethz.ch/data/CINA.html`, 2008.

[15] Chang, C.C. , Lin, C.J., LIBSVM : a library for support vector machines, ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011 `http://www.csie.ntu.edu.tw/ cjlin/libsvm`.

[16] Conn, A.R., Gould, N.I.M., Toint, Ph.L., Trust-region methods, SMPS/SIAM Series on Optimization, 2000.

[17] Deng G., Ferris, M. C., Variable-Number Sample Path Optimization, Mathematical Programming, 117 (1-2), 81-109, 2009.

[18] Dennis, J.E., Schnabel, R.B., Numerical methods for unconstrained optimization and nonlinear equations, Prentice Hall, Englewood Cliffs, NJ, 1983.

[19] Erdogdu M. A., Montanari A., Convergence rates of sub-sampled Newton methods, NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems, 2, 3052-3060, 2015.

[20] Friedlander M.P.,Schmidt M., Hybrid deterministic-stochastic methods for data fitting, SIAM Journal on Scientific Computing, 34(3), 1380-1405, 2012.

[21] Golub, G and Van Loan, C, Matrix Computation, The Johns Hopkins University Press, 1996.

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

[23] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278-2324, 1998. MNIST database available at `http://yann.lecun.com/exdb/mnist/`.

[24] Lichman M., UCI machine learning repository, `https://archive.ics.uci.edu/ml/index.php`, 2013.

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

[26] Krejić, N., Martínez, J.M., Inexact Restoration approach for minimization with inexact evaluation of the objective function, Mathematics of Computation, 85 (2016), 1775-1791.

[27] Krejić N., Krklec N., Line search methods with variable sample size for unconstrained optimization, Journal of Computational and Applied Mathematics 245, 213-231, 2013.

[28] Krejić N., Krklec Jerinkić N., Nonmonotone line search methods with variable sample size, Numerical Algorithms 68(4), 711-739, 2015.

[29] Krejić, N., Martínez, J.M., Inexact Restoration approach for minimization with inexact evaluation of the objective function, Mathematics of Computation, 85 (2016), 1775-1791.

[30] Martínez, J. M., Inexact restoration method with Lagrangian tangent decrease and new merit function for nonlinear programming. Journal of Optimization Theory and Applications 111, pp. 39-58, 2001.

[31] Martínez, J. M., Pilotta, E. A., Inexact restoration algorithms for constrained optimization, Journal of Optimization Theory and Applications 104, pp. 135-163, 2000.

[32] Nocedal, J., Wright, S. J., Numerical Optimization, Springer Series in Operations Research, Springer, 1999.

[33] Pasupathy R., On Choosing Parameters in Retrospective-Approximation Algorithms for Stochastic Root Finding and Simulation Optimization, Operations Research 58(4), pp. 889-901, 2010.

[34] Pilanci M., Wainwright M. J., Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence, SIAM Journal on Optimization 27(1), 205-245, 2017.

[35] Polak E., Royset J.O., Efficient sample sizes in stochastic nonlinear programing, Journal of Computational and Applied Mathematics 217(2), 301-310, 2008.

[36] Roosta-Khorasani, F., Mahoney M.W., Sub-sampled Newton methods, Mathematical Programming, 174, 293-326, 2019.

[37] Xu P., Yang J., Roosta-Khorasani F., Ré C., and Mahoney M.W., Sub-sampled Newton methods with non-uniform sampling, Advances in Neural Information Processing Systems 30 (NIPS), 2530-2538, 2016.

[38] Xu, P., Roosta-Khorasani, F., Mahoney, M. W., Newton-type methods for non-convex optimization under inexact Hessian information, Mathematical Programming, Mathematical Programming, 2019, https://doi.org/10.1007/s10107-019-01405-z