A stochastic first-order trust-region method with inexact restoration for finite-sum minimization

Stefania Bellavia1 · Nataša Krejić2 · Benedetta Morini1 · Simone Rebegoldi1

Received: 6 October 2021 / Accepted: 19 October 2022 / Published online: 3 November 2022 © The Author(s) 2022

Abstract
We propose a stochastic first-order trust-region method with inexact function and gradient evaluations for solving finite-sum minimization problems. Using a suitable reformulation of the given problem, our method combines the inexact restoration approach for constrained optimization with the trust-region procedure and random models. Differently from other recent stochastic trust-region schemes, our proposed algorithm improves feasibility and optimality in a modular way. We provide the expected number of iterations for reaching a near-stationary point by imposing some probability accuracy requirements on random functions and gradients which are, in general, less stringent than the corresponding ones in literature. We validate the proposed algorithm on some nonconvex optimization problems arising in binary classification and regression, showing that it performs well in terms of cost and accuracy, and allows to reduce the burdensome tuning of the hyper-parameters involved.

Keywords Finite-sum minimization · Inexact restoration · Trust-region methods · Subsampling · Worst-case iteration complexity

Stefania Bellavia, Nataša Krejić, Benedetta Morini and Simone Rebegoldi have contributed equally to this work.

Benedetta Morini
benedetta.morini@unifi.it

Stefania Bellavia
stefania.bellavia@unifi.it

Nataša Krejić
natasak@uns.ac.rs

Simone Rebegoldi
simone.rebegoldi@unifi.it

1 Dipartimento di Ingegneria Industriale, Università degli Studi di Firenze, Viale G.B. Morgagni 40, 50134 Florence, Italy

2 Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Trg Dositeja Obradovića 4, Novi Sad 21000, Serbia
1 Introduction

In this paper we consider the finite-sum minimization problem

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

where \( N \) is very large and finite and \( \phi_i : \mathbb{R}^n \rightarrow \mathbb{R}, 1 \leq i \leq N \), are continuously differentiable. A number of important problems can be stated in this form, e.g., classification problems in machine learning, data fitting problems, sample average approximations of an objective function given in the form of mathematical expectation. In recent years the need for efficient methods for solving (1) resulted in a large body of literature and a number of methods have been proposed and analyzed, see e.g., the reviews [1–3].

It is common to employ subsampled approximations of the objective function and its derivatives with the aim of reducing the computational cost. Focusing on first-order methods, the stochastic gradient [4] and more contemporary variants like SVRG [5, 6], SAG [7], ADAM [8] and SARAH [9] are widely used for their simplicity and low cost per iteration. They do not call for function evaluations but require tuning the learning rate and further possible hyper-parameters such as the mini-batch size. Since the tuning effort may be very computationally demanding [10], more sophisticated approaches use stochastic linesearch or trust-region strategies to adaptively choose the learning rate, see [1, 10–15]. In this context, function and gradient approximations have to satisfy sufficient accuracy requirements with some probability. This, in turn, in case of approximations via sampling, requires adaptive choices of the sample sizes used.

In a further stream of works, problem (1) is reformulated as a constrained optimization problem and the sample size is computed deterministically using the Inexact Restoration (IR) approach. The IR approach has been successfully combined with either the linesearch strategy [16] or the trust-region strategy [17–19]; in these papers, function and gradient estimates are built with gradually increasing accuracy and averaging on the same sample.

We propose a novel trust-region method with random models based on the IR methodology. In our proposed method, feasibility and optimality are improved in a modular way, and the resulting procedure differs from the existing stochastic trust-region schemes [13, 14, 20–22] in the acceptance rule for the step. We provide a theoretical analysis and give a bound on the expected iteration complexity to satisfy an approximate first-order optimality condition; this calls for accuracy conditions on random gradients that are assumed to hold with some sufficiently large but fixed probability and are, in general, less stringent than the corresponding ones in [13, 14, 20–22]. Our theoretical analysis improves over the one for the stochastic trust-region method with inexact restoration given in [19], since we no longer rely on standard theory for deterministic unconstrained optimization invoked eventually when functions and gradients are computed exactly.

The paper is organized as follows. In Sect. 2 we give an overview of random models employed in the trust-region framework and introduce the main features of our contribution. The new algorithm is proposed in Sect. 3 and studied theoretically with respect to the iteration complexity analysis. Extensive numerical results are presented in Sect. 4.
2 Trust-region method with random models

Variants of the standard trust-region method based on the use of random models have been presented, to our knowledge, in [13, 14, 19–23]. They consist in the adaptation of the trust-region framework to the case where random estimates of the derivatives are introduced and function values are either computed exactly [20] or replaced by stochastic estimates [13, 14, 19, 21–23].

The computation and acceptance of the iterates parallel the standard trust-region mechanism, and the success of the procedure relies on function values and models being sufficiently accurate with fixed and large enough probability. The accuracy requests in the mentioned works show many similarities; here we illustrate some issues related to the works [13, 14, 22], which are closer to our approach.

Let $\| \cdot \|$ denote the 2-norm throughout the paper. At iteration $k$ of a first-order stochastic trust-region model, given $x_k$, the positive trust-region radius $\delta_k$ and a random approximation $g_k$ of $\nabla f_N(x_k)$, let consider the model $f_N$ on $B(x_k, \delta_k) = \{ x \in \mathbb{R}^n : \| x - x_k \| \leq \delta_k \}$ and the trust-region problem

$$\min_{\| p \| \leq \delta_k} \zeta_k(x_k + p) = f_N(x_k) + g_k^T p$$

for $f_N$ on $B(x_k, \delta_k) = \{ x \in \mathbb{R}^n : \| x - x_k \| \leq \delta_k \}$ and the trust-region problem $\min_{\| p \| \leq \delta_k} \zeta_k(x_k + p)$. Thus, the trust region step takes the form $p_k = -\delta_k g_k / \| g_k \|$.

Two estimates $f^{k,0}$ and $f^{k,p}$ of $f_N$ at $x_k$ and $x_k + p_k$, respectively, are employed to either accept or reject the trial point $x_k + p_k$. The classical ratio between the actual and predicted reduction is replaced by

$$\rho_k = \frac{f^{k,0} - f^{k,p}}{\zeta_k(x_k) - \zeta_k(x_k + p_k)},$$

and a successful iteration is declared when $\rho_k \geq \eta_1$ and $\| g_k \| \geq \eta_2 \delta_k$ for some constants $\eta_1 \in (0, 1)$ and positive and possibly large $\eta_2$. Note that the computation of both the step $p_k$ and the denominator in (2) are independent of $f_N(x_k)$. Furthermore, note that a successful iteration might not yield an actual reduction in $f_N$ because the quantities involved in $\rho_k$ are random approximations to the true value of the objective function.

The condition $\| g_k \| \geq \eta_1 \delta_k$ is not typical of deterministic trust-region methods for smooth optimization and depends on the fact that $\delta_k$ controls the accuracy of function and gradients. Specifically, the models used are required to be sufficiently accurate with some probability. The model $\zeta_k$ is supposed to be, $\pi_M$-probabilistically, a $\kappa_\ast$-fully linear model of $f_N$ on the ball $B(x_k, \delta_k)$, i.e., the requirement

$$|f_N(y) - \zeta_k(y)| \leq \kappa_\ast \delta_k^2, \quad \| \nabla f_N(y) - g_k \| \leq \kappa_\ast \delta_k, \quad y \in B(x_k, \delta_k)$$

with $\kappa_\ast > 0$, has to be fulfilled at least with probability $\pi_M \in (0, 1)$. Moreover, the estimates $f^{k,0}$ and $f^{k,p}$ are supposed to be $\pi_f$-probabilistically $\epsilon_f$-accurate estimates of $f_N(x_k)$ and $f_N(x_k + p_k)$, i.e., the requirement
\[ |f^{k,0} - f_N(x_k)| \leq \epsilon_F \delta_k^2, \quad |f^{k,s} - f_N(x_k + p_k)| \leq \epsilon_F \delta_k^2, \quad (4) \]

has to be fulfilled at least with probability \( \pi_f \in (0,1) \). Clearly, if \( f_N \) is computed exactly then condition (4) is trivially satisfied.

Convergence analysis in [13, 14, 22] shows that for \( \pi_M \) and \( \pi_f \) sufficiently large it holds \( \lim_{k \to \infty} \delta_k = 0 \) almost surely. Moreover, if \( f_N \) is bounded from below and \( \nabla f_N \) is Lipschitz continuous, then \( \lim_{k \to \infty} ||\nabla f_N(x_k)|| = 0 \) almost surely. Interestingly, the accuracy in (3) and (4) increases as the trust region radius gets smaller but the probabilities \( \pi_M \) and \( \pi_f \) are fixed.

For problem (1) it is straightforward to build approximations of \( f_N \) and \( \nabla f_N \) by sample average approximations

\[
f_M(x) = \frac{1}{M} \sum_{i \in I_M} \phi_i(x), \quad \nabla f_S(x) = \frac{1}{S} \sum_{i \in I_S} \nabla \phi_i(x), \quad (5)\]

where \( I_M \) and \( I_S \) are subsets of \( \{1, \ldots, N\} \) of cardinality \(|I_M| = M \) and \(|I_S| = S \), respectively. The choice of sample size such that (3) and (4) hold in probability is discussed in [14, §5] as follows. Let \( \mathbb{E}_\omega[|\phi_\omega(x) - f_N(x)|^2] \leq V_f, \quad \mathbb{E}_\omega[||\nabla \phi_\omega(x) - \nabla f_N(x)||^2] \leq V_g, \quad \forall x \in \mathbb{R}^n, \) with \( \mathbb{E}_\omega \) being the expected value with respect to the random index \( \omega \) employed for sampling, and assume

\[
M \geq \frac{V_f}{\epsilon_F^2 (1 - \pi_f) \delta_k^4}, \quad S \geq \frac{V_g}{\kappa_F^2 (1 - \pi_g) \delta_k^2} \quad \text{and} \quad \max\{M, S\} \leq N. \quad (6)
\]

Then \( f^{k,0} \) and \( f^{k,s} \) built as in (5) with sample size \( M \) satisfy (4) with probability \( p_f \), while \( g_k \) built as in (5) with sample size \( S \) satisfies \( ||\nabla f_N(x_k) - g_k|| \leq \kappa_F \delta_k \) with probability \( p_g \). Furthermore, using Taylor expansion and Lipschitz continuity of \( \nabla f_N \), it can be proved that (3) is met with probability \( \pi_M = \pi_f \pi_p g^* \); consequently, a \( \kappa_F \)-fully linear model of \( f_N \) in \( B(x_k, \delta_k) \) is obtained.

In principle, conditions (3), (4) and \( \lim_{k \to \infty} \delta_k = 0 \) imply that \( f^{k,0}, f^{k,s} \) and \( g_k \) will be computed at full precision for \( k \) sufficiently large. On the other hand, in applications such as machine learning, reaching full precision is unlikely since \( N \) is very large and termination is based on the maximum allowed computational effort or on the validation error.

### 2.1 Our contribution

We propose a trust-region procedure with random models based on (5) and combine it with the inexact restoration (IR) method for constrained optimization [24]. To this end, we make a simple transformation of (1) into a constrained problem. Specifically, letting \( I_M \) be an arbitrary nonempty subset of \( \{1, \ldots, N\} \) of cardinality \(|I_M|\) equal to \( M \), we reformulate problem (1) as

\[ \mathbb{S} \]
A stochastic first-order trust-region method with inexact...  

Using the IR strategy allows to improve feasibility and optimality in a modular way and gives rise to a procedure that differs from the existing trust-region schemes in the following respects. First, at each iteration a reference sample size is fixed and used as a guess for the approximation of function values. Second, the acceptance rule for the step is based on the condition $\|g_k\| \geq \eta_2 \delta_k$, for some $\eta_2 > 0$, and a sufficient decrease condition on a merit function that measures both the reduction of the objective function and the improvement in feasibility. Finally, the expected iteration complexity to satisfy an approximate first-order optimality condition is given, provided that, at each iteration $k$, the gradient estimates satisfy accuracy requirements of order $O(\delta_k)$; such accuracy requirements implicitly govern function approximations and are, in general, less stringent than the corresponding ones in [13, 14, 20–22], as carefully detailed in Sect. 3.

Our theoretical analysis improves over the analysis carried out in [19] for a similar stochastic trust-region coupled with inexact restoration, since here we do not rely on the occurrence of full precision, $M = N$ in (7), reached eventually and do not apply standard theory for unconstrained optimization. In fact, the expected number of iterations until a prescribed accuracy is reached is provided without invoking full precision.

3 The algorithm

In this section we introduce our new algorithm referred to as SIRTR (Stochastic Inexact Restoration Trust Region).

First, we introduce some issues of IR methods. The level of infeasibility with respect to the constraint $M = N$ in (7) is measured by the following function $h$.

**Assumption 1** Let $h : \{1, 2, \ldots, N\} \rightarrow \mathbb{R}$ be a monotonically decreasing function such that $h(1) > 0$, $h(N) = 0$.

This assumption implies that there exist some positive $\underline{h}$ and $\bar{h}$ such that

$$\underline{h} \leq h(M) \text{ if } 1 \leq M < N, \quad \text{and} \quad h(M) \leq \bar{h} \text{ if } 1 \leq M \leq N. \tag{8}$$

One possible choice is $h(M) = (N - M)/N$, $1 \leq M \leq N$.

The IR methods improve feasibility and optimality in modular way using a merit function to balance the progress. Since the reductions in the objective function and infeasibility might be achieved to a different degree, the IR method employs the merit function

$$\Psi(x, M, \theta) = \theta f_M(x) + (1 - \theta) h(M), \tag{9}$$

with $\theta \in (0, 1)$. 

© Springer
Our SIRTR algorithm is a trust-region method that employs first-order random models. At a generic iteration $k$, we fix a trial sample size $N'_{k+1}$ and build a linear model $m_k(p)$ around $x_k$ of the form

$$m_k(p) = f_{N'_{k+1}}(x_k) + g_k^T p,$$

where $g_k$ is a random estimator to $\nabla f_N(x_k)$. Then, we consider the trust-region problem

$$\min_{\|p\| \leq \delta_k} m_k(p),$$

whose solution is

$$p_k = -\delta_k \frac{g_k}{\|g_k\|}.$$

As in standard trust-region methods, we distinguish between successful and unsuccessful iterations. However, we do not employ here the classical acceptance condition, but a more elaborate one that involves the merit function (9).

The proposed method is sketched in Algorithm 3.1 and its steps are now discussed. At a generic iteration $k$, we have at hand the outcome of the previous iteration: the iterate $x_k$, the sample sizes $N_k$ and $\tilde{N}_k$, the penalty parameter $\theta_k$, the flag iflag. If iflag=succ the previous iteration was successful, i.e., $x_k = x_{k-1} + p_{k-1}$, if iflag=unsucc the previous iteration was unsuccessful, i.e., $x_k = x_{k-1}$.

The scheduling procedure for generating the trial sample size $N'_{k+1}$ consists of Steps 1 and 2 of SIRTR. At Step 1, we determine a reference sample size $\tilde{N}_{k+1} \leq N$. If iflag=succ, then the infeasibility measure $h$ is sufficiently decreased as stated in (20). If iflag=unsucc, $\tilde{N}_{k+1}$ is left unchanged from the previous iteration, i.e., $\tilde{N}_{k+1} = \tilde{N}_k$. We remark that (20) trivially implies $\tilde{N}_{k+1} = N$ if $N_k = N$ and that it holds at each iteration, even when it is not explicitly enforced at Step 1 (see forthcoming Lemma 1). In principle $\tilde{N}_{k+1}$ could be the trial sample size but we aim at giving more freedom to the sample size selection process. Thus, at Step 2, we choose a trial sample size $N'_{k+1}$ complying with condition (21). On the one hand, such a condition allows the choice $N'_{k+1} < \tilde{N}_{k+1}$ in order to reduce the computational effort; on the other hand, the choice $N'_{k+1} \geq \tilde{N}_{k+1}$ is also possible in order to satisfy specific accuracy requirements that will be specified later. When $N'_{k+1} < \tilde{N}_{k+1}$, condition (21) rules the largest possible distance between $N'_{k+1}$ and $\tilde{N}_{k+1}$ in terms of $\delta_k$; in case $N'_{k+1} \geq \tilde{N}_{k+1}$, (21) is trivially satisfied.

At Step 3 we form the linear random model (10) and compute its minimizer within the trust-region. Specifically, we fix the cardinality $N_{k+1,g}$ and choose the set of indices $I_{N_{k+1,g}} \subseteq \{1, \ldots, N\}$ of cardinality $N_{k+1,g}$. Then, we compute the estimator $g_k$ of $\nabla f_N(x_k)$ as
A stochastic first-order trust-region method with inexact…

and the solution \( p_k \) in (12) of the trust-region subproblem (11). Further, we compute \( m_k(p_k) \) where \( m_k \) is defined in (10) and

\[
f_{N_{k+1}'}(x_k) = \frac{1}{N_{k+1}'} \sum_{i \in I_{k+1}'} \phi_i(x_k). \tag{14}
\]

with \( I_{k+1}' \subseteq \{1, \ldots, N\} \) being a set of cardinality \( N_{k+1}' \).

At Step 4 we compute the new penalty term \( \theta_{k+1} \). The computation relies on the predicted reduction defined as

\[
\text{Pred}_k(\theta) = \theta(f_{N_k}(x_k) - m_k(p_k)) + (1 - \theta)(h(N_k) - h(\tilde{N}_{k+1})), \tag{15}
\]

where \( \theta \in (0, 1) \). This predicted reduction is a convex combination of the usual predicted reduction \( f_{N_k}(x_k) - m_k(p_k) \) in trust-region methods, and the predicted reduction \( h(N_k) - h(\tilde{N}_{k+1}) \) in infeasibility obtained in Step 1. The new parameter \( \theta_{k+1} \) is computed so that

\[
\text{Pred}_k(\theta) \geq \eta_1(h(N_k) - h(\tilde{N}_{k+1})). \tag{16}
\]

If (16) is satisfied at \( \theta = \theta_k \) then \( \theta_{k+1} = \theta_k \), otherwise \( \theta_{k+1} \) is computed as the largest value for which the above inequality holds (see forthcoming Lemma 2).

Step 5 establishes if the iteration is successful or not. To this end, given a point \( \hat{x} \) and \( \theta \in (0, 1) \), the actual reduction of \( \Psi \) at the point \( \hat{x} \) has the form

\[
\text{Ared}_k(\hat{x}, \theta) = \Psi(x_k, N_k, \theta) - \Psi(\hat{x}, N_{k+1}', \theta) = \theta(f_{N_k}(x_k) - f_{N_{k+1}'}(\hat{x})) + (1 - \theta)(h(N_k) - h(N_{k+1}')), \tag{17}
\]

and the iteration is successful whenever the following two conditions are both satisfied

\[
\text{Ared}_k(x_k + p_k, \theta_{k+1}) \geq \eta_1 \text{Pred}_k(\theta_{k+1}) \tag{18}
\]

\[
\|g_k\| \geq \eta_2 \delta_k. \tag{19}
\]

Otherwise the iteration is declared unsuccessful. If the iteration is successful, we accept the step and the trial sample size, set \( \text{iflag} = \text{succ} \) and possibly increase the trust-region radius through (23); the upper bound \( \delta_{\text{max}} \) on the trust region size is imposed in (23). In case of unsuccessful iterations, we reject both the step and the trial sample size, set \( \text{iflag} = \text{unsucc} \) and decrease the trust region size.
Concerning conditions (18) and (19), we observe that the former mimics the classical acceptance criterion of standard trust-region methods while the latter drives $\delta_k$ to zero as $\|g_k\|$ tends to zero.

\textbf{Algorithm 3.1: The Stochastic IRTR algorithm}

Given $x_0 \in \mathbb{R}^n$, $N_0$ integer in $(0, N]$, $\theta_0 \in (0, 1)$, $0 < \delta_0 < \delta_{\text{max}}$, $\gamma > 1$, $r \in (0, 1)$, $\eta_1 \in (0, 1)$, $\mu > 0$, $\eta_2 > 0$.

0. Set $k = 0$, iflag=succ.
1. If iflag=succ
   Find $\bar{N}_{k+1}$ such that $N_k \leq \bar{N}_{k+1} \leq N$ and
   \[ h(\bar{N}_{k+1}) \leq \tau h(N_k), \] (20)

   Else set $\bar{N}_{k+1} = \bar{N}_k$.
2. If $N_k = N$ set $N_{k+1} = N$
   Else find $N_{k+1}^t$ such that
   \[ h(N_{k+1}^t) - h(\bar{N}_{k+1}) \leq \mu \delta_k^2, \] (21)

3. Choose $N_{k+1,\rho}, I_{N_{k+1,\rho}} \subseteq \{1, \ldots, N\}$ s.t. $|I_{N_{k+1,\rho}}| = N_{k+1,\rho}$.
   Compute $g_k$ as in (13), and set
   \[ p_k = -\delta_k \frac{g_k}{\|g_k\|}. \]
   Compute $f_{N_{k+1}^t}(x_k)$ as in (14), and $m_k(p_k) = f_{N_{k+1}^t}(x_k) + g_k^T p_k$.
4. Compute the penalty parameter $\theta_{k+1}$
   \[ \theta_{k+1} = \begin{cases} 
   \theta_k & \text{if } \text{Pred}_k(\theta_k) \geq \eta_1 (h(N_k) - h(\bar{N}_{k+1})) \\
   \frac{(1 - \eta_1)(h(N_k) - h(\bar{N}_{k+1}))}{m_k(p_k) - f_{N_k}(x_k) + h(N_k) - h(\bar{N}_{k+1})} & \text{otherwise.} \end{cases} \] (22)
5. If $\text{Ared}_k(x_k + p_k, \theta_{k+1}) \geq \eta_1 \text{Pred}_k(\theta_{k+1})$ and $\|g_k\| \geq \eta_2 \delta_k$
   \textbf{(successful iteration)} define
   \[ x_{k+1} = x_k + p_k \]
   \[ \delta_{k+1} = \min \{ \gamma \delta_k, \delta_{\text{max}} \} \] (23)
   set $N_{k+1} = N_{k+1}^t, k = k + 1$, iflag=succ and go to Step 1.
   Else \textbf{(unsuccessful iteration)} define
   \[ x_{k+1} = x_k \]
   \[ \delta_{k+1} = \frac{\delta_k}{\gamma}, \] (24)
   set $N_{k+1} = N_k, k = k + 1$, iflag=unsucc and go to Step 1.

We conclude the description of Algorithm 3.1 showing that condition (20) holds for all iterations, even when it is not explicitly enforced at Step 1.
Lemma 1 Let Assumption 3.1 holds and \( r \in (0, 1) \) be the scalar in Algorithm 3.1. The sample sizes \( \tilde{N}_{k+1} \leq N \) and \( N_k \leq N \) generated by Algorithm 3.1 satisfy

\[
h(\tilde{N}_{k+1}) \leq rh(N_k), \quad \forall k \geq 0. \tag{25}
\]

**Proof** We observe that, by Assumption 3.1, (25) trivially holds whenever \( N_k = \tilde{N}_{k+1} = N \).

Otherwise, we proceed by induction. Indeed, the thesis trivially holds for \( k = 0 \), as we set \( \text{iflag} = \text{succ} \) at the first iteration and enforce (25) at Step 1. Now consider a generic iteration \( \tilde{k} \geq 1 \) and suppose that (25) holds for \( \tilde{k} - 1 \). If iteration \( \tilde{k} - 1 \) is successful, then condition (25) holds also at iteration \( \tilde{k} \).

If iteration \( \tilde{k} - 1 \) is unsuccessful, then at Step 5 we set \( \tilde{N}_{\tilde{k}} = \tilde{N}_{\tilde{k}-1} \). Successively, at Step 1 of iteration \( \tilde{k} \) we set \( \tilde{N}_{\tilde{k}+1} = \tilde{N}_{\tilde{k}} \). Since (25) holds by induction at iteration \( \tilde{k} - 1 \), we have \( h(\tilde{N}_{\tilde{k}}) \leq rh(N_{\tilde{k}-1}) \), which can be rewritten as \( h(\tilde{N}_{\tilde{k}+1}) \leq rh(N_{\tilde{k}}) \) due to the previous assignments at Step 5 and Step 1. Then condition (25) holds also at iteration \( \tilde{k} \).

\( \square \)

### 3.1 On the sequences \( \{\theta_k\} \) and \( \{\delta_k\} \)

In this section, we analyze the properties of Algorithm 3.1. In particular, we prove that the sequence \( \{\theta_k\} \) is non increasing and uniformly bounded from below, and that the trust region radius \( \delta_k \) tends to 0 as \( k \to \infty \). We make the following assumption.

**Assumption 2** Functions \( \phi_i \) are continuously differentiable for \( i = 1, \ldots, n \). There exists \( f_{\text{low}} \in \mathbb{R} \) such that

\[
f_M(x) \geq f_{\text{low}}, \quad 1 \leq M \leq N, \ x \in \mathbb{R}^n.
\]

Furthermore, there exist \( \Omega \subset \mathbb{R}^n \) and \( f_{\text{up}} \in \mathbb{R} \) such that

\[
f_M(x) \leq f_{\text{up}}, \quad 1 \leq M \leq N, \ x \in \Omega,
\]

and all iterates generated by Algorithm 3.1 belong to \( \Omega \).

In the following, we let

\[
k_{\phi} = \max \{|f_{\text{low}}|, |f_{\text{up}}|\}. \tag{26}
\]

**Remark 1** In the context of machine learning, the above assumption is verified in several cases, e.g., the mean-squares loss function coupled with either the sigmoid, the softmax or the hyperbolic tangent activation function; the mean-squares loss function coupled with ReLU or ELU activation functions and bound constraints (above and below) on all variables; the logistic loss function coupled again with bound constraints (above and below) on the unknowns [25].
In the analysis that follows we will consider two options for \( \hat{x} \) in (17), \( \hat{x} = x_k + p_k \) for successful iterations and \( \hat{x} = x_k \) for unsuccessful iterations.

Our first result characterizes the sequence \( \{\theta_k\} \) of the penalty parameters; the proof follows closely [19, Lemma 2.2].

**Lemma 2** Let Assumptions 1 and 2 hold. Then the sequence \( \{\theta_k\} \) is positive, non increasing and bounded from below, \( \theta_{k+1} \geq \theta > 0 \) with \( \theta \) independent of \( k \) and (16) holds with \( \theta = \theta_{k+1} \).

**Proof** We note that \( \theta_0 > 0 \) and proceed by induction assuming that \( \theta_k \) is positive. Due to Lemma 1, for all iterations \( k \) we have that \( N_k \leq \tilde{N}_{k+1} \) and that \( N_k = \tilde{N}_{k+1} \) if and only if \( N_k = N \). First consider the case where \( N_k = \tilde{N}_{k+1} \) (or equivalently \( N_k = \tilde{N}_{k+1} = N \)); then it holds \( h(N_k) - h(\tilde{N}_{k+1}) = 0 \), and \( N_{k+1}' = N \) by Step 2. Therefore, we have

\[
\text{Pred}_k(\theta) = \theta \delta_k \|g_k\| > 0 \quad \text{for any positive } \theta, \quad \text{and (22) implies } \theta_{k+1} = \theta_k.
\]

Hence, we only need to consider the case \( N_k < \tilde{N}_{k+1} \). If inequality \( \text{Pred}_k(\theta) \geq \eta_1 (h(N_k) - h(\tilde{N}_{k+1})) \) holds then (22) gives \( \theta_{k+1} = \theta_k \). Otherwise, we have

\[
\theta_k \left( f_{N_k}(x_k) - m_k(p_k) - (h(N_k) - h(\tilde{N}_{k+1})) \right) < (\eta_1 - 1) \left( h(N_k) - h(\tilde{N}_{k+1}) \right),
\]

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

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

Consequently, \( \text{Pred}_k(\theta) \geq \eta_1 (h(N_k) - h(\tilde{N}_{k+1})) \) is satisfied if

\[
\theta (f_{N_k}(x_k) - m_k(p_k) - (h(N_k) - h(\tilde{N}_{k+1}))) \geq (\eta_1 - 1)(h(N_k) - h(\tilde{N}_{k+1})),
\]

i.e., if

\[
\theta \leq \theta_{k+1} \overset{\text{def}}{=} \frac{(1 - \eta_1)(h(N_k) - h(\tilde{N}_{k+1}))}{m_k(p_k) - f_{N_k}(x_k) + h(N_k) - h(\tilde{N}_{k+1})}.
\]

Hence \( \theta_{k+1} \) is the largest value satisfying (16) and \( \theta_{k+1} < \theta_k \).

Let us now prove that \( \theta_{k+1} \geq \theta \). Note that by (25) and (8)

\[
h(N_k) - h(\tilde{N}_{k+1}) \geq (1 - r)h(N_k) \geq (1 - r)\bar{h}.
\]

Using (26)

\[
m_k(p_k) - f_{N_k}(x_k) + h(N_k) - h(\tilde{N}_{k+1}) \leq m_k(p_k) - f_{N_k}(x_k) + h(N_k)
\]

\[
\leq f_{N_{k+1}}(x_k) - \delta_k \|g_k\| - f_{N_k}(x_k) + \bar{h}
\]

\[
\leq |f_{N_{k+1}}(x_k) - f_{N_k}(x_k)| + \bar{h} \leq 2k + \bar{h},
\]

Springer
and \( \theta_{k+1} \) in (22) satisfies
\[
\theta_{k+1} \geq \theta = \frac{(1 - \eta_1)(1 - r)h}{2k_{\phi} + \bar{h}},
\]
which completes the proof.

In the following, we derive bounds for the actual reduction \( \text{Ared}_k(x_{k+1}, \theta_{k+1}) \) in case of successful iterations and distinguish the iteration indexes \( k \) as below:
\[
\mathcal{I}_1 = \{ k \geq 0 \text{ s.t. } N_k < \bar{N}_{k+1} \},
\]
\[
\mathcal{I}_2 = \{ k \geq 0 \text{ s.t. } N_k = \bar{N}_{k+1} \}.
\]
Note that \( \mathcal{I}_1, \mathcal{I}_2 \) are disjoint and any iteration index \( k \) belongs to exactly one of these subsets. Moreover, (25) yields \( \bar{N}_{k+1} = N_k = N'_{k+1} = N_{k+1} \) for any \( k \in \mathcal{I}_2 \).

**Lemma 3** Let Assumptions 1-2 hold and suppose that iteration \( k \) is successful. If \( k \in \mathcal{I}_1 \) then
\[
\text{Ared}_k(x_{k+1}, \theta_{k+1}) \geq \frac{\eta_1^2(1 - r)h}{\delta_{\max}^2} \delta_k^2.
\]
Otherwise,
\[
\text{Ared}_k(x_{k+1}, \theta_{k+1}) \geq \eta_1 \eta_2 \theta \delta_k^2.
\]

**Proof** Since iteration \( k \) is successful, \( x_{k+1} = x_k + p_k \) and (18) hold. Suppose \( k \in \mathcal{I}_1 \). By (18) and (16)
\[
\text{Ared}_k(x_k + p_k, \theta_{k+1}) \geq \eta_1 \text{Pred}_k(\theta_{k+1}) \geq \eta_1^2 (h(N_k) - h(\bar{N}_{k+1})).
\]
In virtue of Lemma 1 we have \( h(N_k) - h(\bar{N}_{k+1}) \geq (1 - r)h(N_k) \), hence we obtain
\[
\text{Ared}_k(x_k + p_k, \theta_{k+1}) \geq \eta_1^2 (1 - r)h(N_k).
\]
Dividing and multiplying the right-hand side above by \( \delta_k^2 \), applying the inequalities \( h \leq h(N_k), \delta_k \leq \delta_{\max} \), we get (31).

Suppose \( k \in \mathcal{I}_2 \). Then \( N_k = \bar{N}_{k+1} \) and by the definition of \( \text{Pred}_k(\theta_{k+1}) \) and Lemma 2, we have
\[
\text{Pred}_k(\theta_{k+1}) = \theta_{k+1}(f_N(x_k) - m_k(p_k)) = \theta_{k+1} \delta_k \| g_k \| \geq \theta \delta_k \| g_k \|,
\]
and therefore (18), (19) and Lemma 2 yield (32).

Let us now define a Lyapunov type function \( \Phi \) inspired by the paper [14]. Assumption 1 implies that \( h(N_k) \) is bounded from above while Assumption 2 implies that \( f_N(x) \) is bounded from below if \( x \in \Omega \). Thus, there exists a constant \( \Sigma \) such that
\[ f_{N_k}(x) - h(N_k) + \Sigma \geq 0, \quad x \in \Omega, \quad k \geq 0. \] (33)

**Definition 1**  
Let \( v \in (0, 1) \) be a fixed constant. For all \( k \geq 0 \), we define

\[
\phi_k \overset{\text{def}}{=} \Phi(x_k, N_k, \theta_k, \delta_k) = v(\Psi(x_k, N_k, \theta_k) + \theta_k \Sigma) + (1 - v)\delta_k^2,
\] (34)

where \( \Psi \) is the merit function given in (9) and \( \Sigma \) is given in (33).

The choice of \( v \in (0, 1) \) in the above definition will be specified below. First, note that \( \phi_k \) is bounded below for all \( k \geq 0 \),

\[
\phi_k \geq v(\Psi(x_k, N_k, \theta_k) + \theta_k \Sigma) \\
\geq v(\theta_k f_{N_k}(x_k) + (1 - \theta_k)h(N_k) + \theta_k(-f_{N_k}(x_k) + h(N_k))) \\
\geq vh(N_k) \geq 0.
\] (35)

Second, adding and subtracting suitable terms, by the definition (34) and for all \( k \geq 0 \), we have

\[
\phi_{k+1} - \phi_k = v\left(\theta_{k+1} f_{N_{k+1}}(x_{k+1}) + (1 - \theta_{k+1})h(N_{k+1})\right) \\
- v(\theta_k f_{N_k}(x_k) + (1 - \theta_k)h(N_k)) + v(\theta_{k+1} - \theta_k)\Sigma \\
+ (1 - v)(\delta_{k+1}^2 - \delta_k^2) \\
= v\left(\theta_{k+1} f_{N_{k+1}}(x_{k+1}) + (1 - \theta_{k+1})h(N_{k+1})\right) \\
\pm v(1 - \theta_{k+1})h(N_{k+1}) \\
- v(\theta_k f_{N_k}(x_k) + (1 - \theta_k)h(N_k)) + v(\theta_{k+1} - \theta_k)\Sigma \\
+ (1 - v)(\delta_{k+1}^2 - \delta_k^2) \\
= v\left(\theta_{k+1} f_{N_{k+1}}(x_{k+1}) - f_{N_k}(x_k)\right) + (1 - \theta_{k+1})(h(N_{k+1}) - h(N_k)) \\
+ v(\theta_{k+1} - \theta_k)(f_{N_k}(x_k) - h(N_k) + \Sigma) + (1 - v)(\delta_{k+1}^2 - \delta_k^2).
\] (36)

If the iteration \( k \) is successful, then using (33), the monotonicity of \( \{\theta_k\}_{k \in \mathbb{N}} \) proved in Lemma 2, and the fact that \( N_{k+1} = N'_{k+1} \), the equality (36) yields

\[
\phi_{k+1} - \phi_k \leq -vA_{\text{red}}(x_{k+1}, \theta_{k+1}) + (1 - v)(\delta_{k+1}^2 - \delta_k^2).
\] (37)

Otherwise, if the iteration \( k \) is unsuccessful, then \( x_{k+1} = x_k, N_{k+1} = N_k \) and thus the first quantity at the right-hand side of equality (36) is zero. Hence using again (33) and the monotonicity of \( \{\theta_k\}_{k \in \mathbb{N}} \), we obtain

\[
\phi_{k+1} - \phi_k \leq (1 - v)(\delta_{k+1}^2 - \delta_k^2).
\] (38)

Now we provide bounds for the change of \( \Phi \) along subsequent iterations and again distinguish the two cases \( k \in I_1, I_2 \) stated in (29)-(30).
Lemma 4 Let Assumptions 1 and 2 hold.

(i) If the iteration $k$ is unsuccessful, then

$$\phi_{k+1} - \phi_k \leq \chi_1 \delta_k^2, \quad \chi_1 = (1 - \nu) \frac{1 - \gamma^2}{\gamma^2}. \quad (39)$$

(ii) If the iteration $k$ is successful and $k \in I_1$, then

$$\phi_{k+1} - \phi_k \leq \chi_2 \delta_k^2, \quad \chi_2 = \left( -\nu \left( \frac{\eta_1^2 (1 - r) h}{\delta_{\max}^2} \right) + (1 - \nu)(\gamma^2 - 1) \right). \quad (40)$$

If the iteration $k$ is successful and $k \in I_2$, then

$$\phi_{k+1} - \phi_k \leq \chi_3 \delta_k^2, \quad \chi_3 = (-\nu \eta_1 \eta_2 \theta + (1 - \nu)(\gamma^2 - 1)). \quad (41)$$

Proof (i) If iteration $k$ is unsuccessful, the updating rule (24) for $\delta_{k+1}$ implies $\delta_{k+1} = \delta_k / \gamma$. Thus, equation (38) directly yields (39).

(ii) If iteration $k$ is successful, the updating rule (23) for $\delta_{k+1}$ implies $\delta_{k+1} \leq \gamma \delta_k$. Thus combining (37) with Lemma 3 we obtain (40) and (41).

We are now ready to prove that a sufficient decrease condition holds for $\Phi$ along subsequent iterations and that $\delta_k$ tends to zero.

Theorem 1 Let Assumptions 1 and 2 hold. There exists $\sigma > 0$, depending on $\nu \in (0, 1)$ in (34), such that

$$\phi_{k+1} - \phi_k \leq -\sigma \delta_k^2, \quad \text{for all } k \geq 0. \quad (42)$$

Proof In case of unsuccessful iterations, (39) provides a sufficient decrease $\phi_{k+1} - \phi_k$ for any value of $\nu \in (0, 1)$. In case of successful iterations, $\chi_2$ and $\chi_3$ in (40) and (41) are both negative if

$$\max \left\{ \frac{(\gamma^2 - 1) \delta_{\max}^2}{\eta_1^2 (1 - r) h + (\gamma^2 - 1) \delta_{\max}^2}, \frac{\gamma^2 - 1}{\eta_1 \eta_2 \theta + \gamma^2 - 1} \right\} < \nu < 1. \quad (43)$$

Therefore, if $\nu$ is chosen as above and

$$\sigma = \min\{\chi_1, \chi_2, \chi_3\}, \quad (44)$$

then (39)–(41) imply (42) and the proof is completed.

Theorem 2 Let Assumptions 1 and 2 hold. Then the sequence $\{\delta_k\}$ in Algorithm 3.1 satisfies
\[
\lim_{k \to \infty} \delta_k = 0.
\]

**Proof** Under the stated conditions Theorem 1 holds and summing up (42) for \( j = 0, 1, \ldots, k - 1 \), we obtain

\[
\phi_k - \phi_0 = \sum_{j=0}^{k-1} (\phi_{j+1} - \phi_j) \leq -\sigma \sum_{j=0}^{k-1} \delta_j^2.
\]

Given that, by (35), \( \phi_k \) is bounded from below for all \( k \), we conclude that \( \sum_{j=0}^{\infty} \delta_j^2 < \infty \), and hence \( \lim_{j \to \infty} \delta_j = 0 \). \( \square \)

### 3.2 Complexity analysis

Algorithm 3.1 generates a random process since the function estimates in (14) and gradient estimates in (13) are random. All random quantities are denoted by capital letters, while the use of small letters is reserved for their realizations. In particular, the iterates \( X_k \), the trust region radius \( \Delta_k \), the steps \( P_k \), the function estimates \( F_{N_{k+1}}(X_k), F_{N_{k+1}}(X_k + P_k) \), the gradient estimates \( G_k, \nabla F_{N_{k+1}}(X_k) \), and the value \( \Phi_k \) of the function \( \Phi \) in (34) at iteration \( k \) are random variables, while \( x_k, \delta_k, p_k, f_{N_{k+1}}(x_k), f_{N_{k+1}}(x_k + p_k), g_k, \nabla f_{N_{k+1}}(x_k), \phi_k \) are their realizations.

In this section, our aim is to derive a bound on the expected number of iterations that occur in Algorithm 3.1 to reach a desired accuracy. We show that our algorithm is included into the stochastic framework given in [13, Section 2] and consequently we derive an upper bound on the expected value of the hitting time \( K_{c_e} \) defined below.

**Definition 2** Given \( \epsilon > 0 \), the hitting time \( K_{c_e} \) is the random variable \( K_{c_e} = \min\{k \geq 0 : \|\nabla f_N(X_k)\| \leq \epsilon\} \), i.e., \( K_{c_e} \) is the first iteration such that \( \|\nabla f_N(X_k)\| \leq \epsilon \).

Our analysis relies on the assumption that \( g_k \) and \( \nabla f_{N_{k+1}}(x_k) \) are probabilistically accurate estimators of the true gradient at \( x_k \), in the sense that the events

\[
G_{k,1} = \{\|\nabla f_N(X_k) - G_k\| \leq \nu \Delta_k\},
\]

\[
G_{k,2} = \{\|\nabla f_N(X_k) - \nabla f_{N_{k+1}}(X_k)\| \leq \nu \Delta_k\},
\]

(45) (46)
are true at least with conditioned probability \( \pi_1 \in (0, 1) \) and \( \pi_2 \in (0, 1) \), respectively. Using the same terminology of [26, 27], we say that iteration \( k \) is true if both \( G_{k,1} \) and \( G_{k,2} \) are true. Furthermore, we introduce the two random variables

\[
I_k = \mathbb{1}_{g_{k,1}}, \quad J_k = \mathbb{1}_{g_{k,2}},
\]

(47)

where \( \mathbb{1}_A \) denotes the indicator function of an event \( A \).

Finally, we need the following additional assumptions.

**Assumption 3** The gradients \( \nabla \phi_i \) are Lipschitz continuous with constant \( L_i \). Let 

\[
L = \frac{1}{2} \max_{1 \leq i \leq N} L_i.
\]

Under Assumptions 2 and 3, the norm of the gradient estimates \( \|g_k\| \) is bounded, as shown below.

**Lemma 5** Let Assumptions 2 and 3 hold. Then there exists \( g_{\max} \) such that

\[
\|g_k\| \leq g_{\max}, \quad k \geq 0,
\]

(48)

where \( g_{\max} = \sqrt{8L\kappa_\phi} \) and \( \kappa_\phi \) is given in (26).

**Proof** By Assumption 3, it easily follows that \( \nabla f_{N+k,g} \) is Lipschitz continuous on \( \mathbb{R}^n \) with constant \( 2L \). Then Assumption 2 and the descent lemma for continuously differentiable functions with Lipschitz continuous gradient [28, Proposition A.24] ensure that

\[
f_{\text{low}} \leq f_{N+k,g}(y) \leq f_{N+k,g}(x) + \nabla f_{N+k,g}(x)^T (y - x) + L\|y - x\|^2, \quad \forall x, y \in \mathbb{R}^n.
\]

Taking the minimum of the right-hand side with respect to \( y \), we can also write

\[
f_{\text{low}} \leq \min_{y \in \mathbb{R}^n} \zeta(y) \equiv f_{N+k,g}(x) + \nabla f_{N+k,g}(x)^T (y - x) + L\|y - x\|^2, \quad \forall x \in \mathbb{R}^n.
\]

The minimum of \( \zeta(y) \) is attained at the point \( \tilde{y} = x - \frac{1}{2L} \nabla f_{N+k,g}(x) \) and letting \( y = \tilde{y} \) in the previous inequality, we get:

\[
f_{\text{low}} \leq f_{N+k,g}(x) - \frac{1}{2L} \|\nabla f_{N+k,g}(x)\|^2 + \frac{1}{4L} \|\nabla f_{N+k,g}(x)\|^2, \quad \forall x \in \mathbb{R}^n,
\]

and equivalently \( \|\nabla f_{N+k,g}(x)\|^2 \leq 4L(f_{N+k,g}(x) - f_{\text{low}}) \) for all \( x \in \mathbb{R}^n \). Using again Assumption 2, we have \( f_{N+k,g}(x) - f_{\text{low}} \leq |f_{N+k,g}(x)| + |f_{\text{low}}| \leq 2\kappa_\phi \), and consequently

\[
\|\nabla f_{N+k,g}(x)\|^2 \leq 8L\kappa_\phi, \quad \forall x \in \mathbb{R}^n.
\]

Setting \( x = x_k \) and \( g_{\max} = \sqrt{8L\kappa_\phi} \) in the previous inequality, we get the result. \( \square \)
First, we analyze the occurrence of successful iterations and show that the availability of accurate gradients has an impact on the acceptance of the trial steps. The following lemma establishes that if the iteration $k$ is true and $\delta_k$ is smaller than a certain threshold, then the iteration is successful. The analysis is presented for a single realization of Algorithm 3.1 and specializes for $k$ in the sets $\mathcal{I}_1, \mathcal{I}_2$.

**Lemma 6** Let Assumptions 1–3 hold and suppose that iteration $k$ is true.

(i) If $k \in \mathcal{I}_1$, then the iteration is successful whenever

$$\delta_k \leq \min \left\{ \frac{\|g_k\|}{\eta_3}, \frac{\|g_k\|}{\eta_2} \right\},$$

where $\eta_3 = \frac{\delta_{\text{max}}(\eta_0(2\nu + L) + (1 - \nu)\mu)}{\eta_1(1 - \eta_1)(1 - \nu)}$.

(ii) If $k \in \mathcal{I}_2$, then the iteration is successful whenever

$$\delta_k \leq \min \left\{ \frac{(1 - \eta_1)\|g_k\|}{2\nu + L}, \frac{\|g_k\|}{\eta_2} \right\}.$$

**Proof** From Assumption 3, it follows that $\nabla f_{N_{k+1}}$ is Lipschitz continuous with constant $2L$. Then,

$$|m_k(p_k) - f_{N_{k+1}}(x_k + p_k)| = \left| \int_0^1 \left( g_k \pm \nabla f_{N_{k+1}}(x_k) - \nabla f_{N_{k+1}}(x_k + \tau p_k) \right)^T p_k d\tau \right|$$

$$\leq \int_0^1 \|g_k - \nabla f_{N_{k+1}}(x_k)\| \|p_k\| d\tau + \int_0^1 2L\tau \|p_k\|^2 d\tau$$

$$\leq \int_0^1 (\|g_k - \nabla f_{N}(x_k)\| + \|\nabla f_N(x_k) - \nabla f_{N_{k+1}}(x_k)\|) \|p_k\| d\tau$$

$$+ \int_0^1 2L\tau \|p_k\|^2 d\tau$$

and, since $G_{k,1}$ and $G_{k,1}$ are both true, (45) and (46) yield

$$|m_k(p_k) - f_{N_{k+1}}(x_k + p_k)| \leq (2\nu + L)\delta_k^2.$$

Now, let us analyze condition (18) for successful iterations.

(i) If $k \in \mathcal{I}_1$, by (15), (17) and (16) we obtain
A red_k(x_k + p_k, \theta_{k+1}) - \eta_1 \text{Pred}_k(\theta_{k+1}) = (1 - \eta_1) \text{Pred}_k(\theta_{k+1}) \\
+ A red_k(\theta_{k+1}) - \text{Pred}_k(\theta_{k+1}) \\
= (1 - \eta_1) \text{Pred}_k(\theta_{k+1}) \\
+ \theta_{k+1}(m_k(p_k) - f^\prime_{N_{k+1}^i}(x_k + p_k)) \\
+ (1 - \theta_{k+1})(h(\tilde{N}_{k+1}) - h(N_{k+1}^i)) \\
\geq \eta_1(1 - \eta_1)(h(N_k) - h(\tilde{N}_{k+1})) \\
+ \theta_{k+1}(m_k(p_k) - f^\prime_{N_{k+1}^i}(x_k + p_k)) \\
+ (1 - \theta_{k+1})(h(\tilde{N}_{k+1}) - h(N_{k+1}^i)). \\
(53)

Using (52), (21) and \theta \leq \theta_{k+1} \leq \theta_0, we also have

\theta_{k+1}(f_{N_{k+1}^i}(x_k + p_k) - m_k(p_k)) + (1 - \theta_{k+1})(h(N_{k+1}^i) - h(\tilde{N}_{k+1})) \\
\leq \theta_0(2v + L) + (1 - \theta)\mu \delta^2_k. \\
(54)

Note that the combination of (25), (8), (23) and Lemma 5, guarantees that

h(N_k) - h(\tilde{N}_{k+1}) \geq (1 - r)h(N_k) \geq \frac{(1 - r)h\delta_k \|g_k\|}{\delta_{\text{max}} g_{\text{max}}}.

(55)

Then, from (53), (54), and (55), we have

A red_k(x_k + p_k, \theta_{k+1}) - \eta_1 \text{Pred}_k(\theta_{k+1}) \geq \frac{\eta_1(1 - \eta_1)(1 - r)h\delta_k \|g_k\|}{\delta_{\text{max}} g_{\text{max}}} \\
- \theta_0(2v + L) + (1 - \theta)\mu \delta^2_k.

Combining this result with (19), the proof is complete.

(ii) Using (15), (17), k \in \mathbb{Z}_2, we have

A red_k(x_k + p_k, \theta_{k+1}) - \eta_1 \text{Pred}_k(\theta_{k+1}) = (1 - \eta_1) \text{Pred}_k(\theta_{k+1}) \\
+ A red_k(\theta_{k+1}) - \text{Pred}_k(\theta_{k+1}) \\
= (1 - \eta_1)\theta_{k+1}\delta_k \|g_k\| \\
+ \theta_{k+1}(m_k(p_k) - f(x_k + p_k)).

Using (52) we get

A red_k(x_k + p_k, \theta_{k+1}) - \eta_1 \text{Pred}_k(\theta_{k+1}) \geq (1 - \eta_1)\theta_{k+1}\delta_k \|g_k\| \\
- \theta_{k+1}(2v + L)\delta^2_k. \\
(56)

Combining the above inequality with (19), we have proved that the iteration is successful whenever (50) holds.

We can now guarantee that a successful iteration k occurs whenever k is true, the prefixed accuracy \epsilon in Definition 2 has not been achieved at k, and \delta_k is below
a certain threshold depending on \( \epsilon \). Again, the result is stated for a single realization of the algorithm.

**Lemma 7** Let Assumptions 1–3 hold. Suppose that \( \| \nabla f_N(x_k) \| > \epsilon \), for some \( \epsilon > 0 \), the iteration \( k \) is true, and

\[
\delta_k < \delta^\dagger := \min \left\{ \frac{\epsilon}{2\nu}, \frac{\epsilon}{2\eta_1}, \frac{\epsilon(1 - \eta_1)}{2(2\nu + L)} \right\}.
\]

(57)

Then, iteration \( k \) is successful.

**Proof** By \( \| \nabla f_N(x_k) \| > \epsilon \), the occurrence of \( G_{k,1} \) and (57), we have

\[
\| g_k - \nabla f_N(x_k) \| \leq \nu \delta_k < \frac{\epsilon}{2},
\]

and this yields \( \| g_k \| \geq \frac{\epsilon}{2} \). Then, Lemma 6 implies that iteration \( k \) is successful. \( \square \)

We now proceed similarly to [13, Section 2] and analyse the random process \( \{ (\Phi_k, \Delta_k, W_k) \}_{k \in \mathbb{N}} \) generated by Algorithm 3.1, where \( \Phi_k \) is the random variable whose realization is given in (34) and \( W_k \) is the random variable defined as

\[
\begin{cases}
W_0 = 1 \\
W_{k+1} = 2 \left( I_k J_k - \frac{1}{2} \right), & k = 0, 1, \ldots
\end{cases}
\]

(58)

Clearly, \( W_k \) takes values \( \pm 1 \). We denote with \( \mathbb{P}_{k-1}(\cdot) \) and \( \mathbb{E}_{k-1}(\cdot) \) the probability and expected value conditioned to the \( \sigma \)-algebra generated by \( F_{N_1}(X_0), \ldots, F_{N_k}(X_{k-1}), \nabla F_{N_1}(X_0), \ldots, \nabla F_{N_k}(X_{k-1}), G_0, \ldots, G_{k-1} \). Then, we can prove the following result.

**Lemma 8** Let Assumptions 1–3 hold, \( \nu \) as in (43), \( \delta^\dagger \) as in (57) and \( K_\epsilon \) as in Definition 2. Suppose there exists some \( j_{\text{max}} \geq 0 \) such that \( \delta_{\text{max}} = \nu^{j_{\text{max}}} \delta_0 \), and \( \delta_0 > \delta^\dagger \).

Assume that the estimators \( G_k \) and \( \nabla f_{N_{k+1}}(X_k) \) are conditionally independent random variables, and the events \( G_{k,1}, G_{k,2} \) occur with sufficiently high probability, i.e.,

\[
\mathbb{P}_{k-1}(G_{k,1}) = \pi_1, \quad \mathbb{P}_{k-1}(G_{k,2}) = \pi_2, \quad \text{and} \quad \pi_3 = \pi_1 \pi_2 > \frac{1}{2}.
\]

(59)

Then,

(i) there exists \( \lambda > 0 \) such that \( \Delta_k \leq \delta_0 e^{2j_{\text{max}}} \) for all \( k \geq 0 \);

(ii) there exists a constant \( \delta_\epsilon = \delta_0 e^{j_{\text{max}}} \) for some \( j_\epsilon \leq 0 \) such that, for all \( k \geq 0 \),

\[
1_{\{K_\epsilon > k\}} \Delta_{k+1} \geq 1_{\{K_\epsilon > k\}} \min \{ \Delta_k e^{j_{\text{max}}} \delta_\epsilon \},
\]

(60)
where $W_{k+1}$ satisfies

$$P_{k}(W_{k+1} = 1) = \pi_3, \quad P_{k}(W_{k+1} = -1) = 1 - \pi_3; \quad (61)$$

(iii) there exists a nondecreasing function $\ell' : [0, \infty) \to (0, \infty)$ and a constant $\Theta > 0$ such that, for all $k \geq 0$,

$$\mathbb{I}_{\{K_c > k\}} \mathbb{E}_{k-1}[\Phi_{k+1}] \leq \mathbb{I}_{\{K_c > k\}} \Phi_k - \mathbb{I}_{\{K_c > k\}} \Theta \ell'(\Delta_k). \quad (62)$$

**Proof** The proof parallels that of [13, Lemma 7].

(i) Since $\delta_{\max} = \gamma^{j_{\max}} \delta_0$, we can set $\lambda = \log(\gamma) > 0$, and the thesis follows from Step 5 of Algorithm 3.1.

(ii) Let us set

$$\delta_e = \frac{\epsilon}{\xi}, \quad \text{where } \xi \geq \max \left\{ 2v, 2\eta_2, 2\eta_3, \frac{2(v + L)}{1 - \eta_1} \right\}, \quad (63)$$

and assume that $\delta_e = \gamma^j \delta_0$, for some integer $j \leq 0$; notice that we can always choose $\xi$ sufficiently large so that this is true. As a consequence, $\Delta_k = \gamma^j \delta_e$ for some integer $j_k$.

When $\mathbb{I}_{\{K_c > k\}} = 0$, inequality (60) trivially holds. Otherwise, conditioning on $\mathbb{I}_{\{K_c > k\}} = 1$, we can prove that

$$\Delta_{k+1} \geq \min \{ \delta_e, \min \{ \delta_{\max}, \gamma \Delta_k \} I_k J_k + \gamma I_k J_k \}. \quad (64)$$

Indeed, for any realization such that $\delta_k > \delta_e$, we have $\delta_k \geq \gamma \delta_e$ and because of Step 5, it follows that $\delta_{k+1} \geq \delta_e$. Now let us consider a realization such that $\delta_k \leq \delta_e$. Since $K_c > k$ and $\delta_e \leq \delta^*$, if $I_k J_k = 1$ (i.e., $k$ is true), then we can apply Lemma 7 and conclude that $k$ is successful. Hence, by Step 5, we have $\delta_{k+1} = \min \{ \delta_{\max}, \gamma \Delta_k \} I_k J_k + \gamma I_k J_k \). If $I_k J_k = 0$, then we cannot guarantee that $k$ is successful; however, again using Step 5, we can write $\delta_{k+1} \geq \gamma I_k J_k \). Combining these two cases, we get (64). If we observe that $\delta_{\max} = \gamma^{j_{\max}} \delta_0 \geq \gamma^{j_{\max}} \delta_0 = \delta_e$, and recall the definition of $W_k$ in (58), then equation (64) easily yields (60). The probabilistic conditions (61) are a consequence of (59).

(iii) The thesis trivially follows from (42) with $\ell'(\Delta) = \Delta^2$ and $\Theta = \sigma$. \hfill \qed

The previous lemma shows that the random process $\{(\Phi_k, \Delta_k, W_k)\}_{k \in \mathbb{N}}$ complies with Assumption 2.1 of [13].

**Theorem 3** Under the assumptions of Lemma 8, we have

$$\mathbb{E}[K_c] \leq \frac{\pi_3}{2\pi_3 - 1} \cdot \frac{\phi_0 \xi^2}{\sigma \epsilon^2} + 1. \quad (65)$$

where $\xi$ is chosen as in (63) and $\sigma$ is given in (44).
Proof The claim follows directly by [13, Theorem 2].

Remark 2 The requirement of (45) and (46) to hold in probability is less stringent than the overall conditions (3) and (4). Analogously to the discussion in Sect. 2, if $\mathbb{E}_\omega[|\nabla f(x) - \nabla f_N(x)|^2] \leq V_g$, then Chebyshev inequality guarantees that events (45) and (46) hold in probability when

$$\frac{V_g}{\nu^2(1 - \pi_1)\delta_k^2} \leq N_{k+1,g} \leq N, \quad \frac{V_g}{\nu^2(1 - \pi_2)\delta_k^2} \leq N_{k+1} \leq N.$$ 

Clearly, $\min\{N_{k+1,g}, N_{k+1}\} = \mathcal{O}(\delta_k^{-2})$ and in general these sample sizes are expected to grow slower than in (6).

Finally, the complexity theory presented improves on [19] where the iteration complexity before reaching full precision $M = N$ in (7) is estimated, and thereafter existing iteration complexity results for trust-region methods applied to (1) are invoked.

4 Numerical experience

In this section, we evaluate the numerical performance of SIRTR on some non-convex optimization problems arising in binary classification and regression.

All the numerical results have been obtained by running MATLAB R2019a on an Intel Core i7-4510U CPU 2.00-2.60 GHz with an 8 GB RAM. For all our tests, we equip SIRTR with $\delta_0 = 1$ as the initial trust-region radius, $\delta_{\max} = 100$, $\gamma = 2$, $\eta = 10^{-1}$, $\eta_2 = 10^{-6}$. Concerning the inexact restoration phase, we borrow the implementation details from [19]. Specifically, the infeasibility measure $h$ and the initial penalty parameter $\theta_0$ are set as follows:

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

The updating rule for choosing $\tilde{N}_{k+1}$ has the form

$$\tilde{N}_{k+1} = \min\{N, \lceil \tilde{c}N_k \rceil\},$$ 

where $1 < \tilde{c} < 2$ is a prefixed constant factor; note that this choice of $\tilde{N}_{k+1}$ satisfies (21) with $r = (N - (\tilde{c} - 1))/N$. At Step 2 the function sample size $N'_{k+1}$ is computed using the rule

$$N'_{k+1} = \begin{cases} 
\lceil \tilde{N}_{k+1} - \mu N\delta_k^2 \rceil, & \text{if } \lceil \tilde{N}_{k+1} - \mu N\Delta_k^2 \rceil \in [N_0, 0.95N] \\
\tilde{N}_{k+1}, & \text{if } \lceil \tilde{N}_{k+1} - \mu N\Delta_k^2 \rceil < N_0 \\
N, & \text{if } \lceil \tilde{N}_{k+1} - \mu N\Delta_k^2 \rceil > 0.95N.
\end{cases}$$ 

Once the set $I_{N'_{k+1}}$ is fixed, the search direction $g_k \in \mathbb{R}^n$ is computed via sampling as in (13) and the sample size $N_{k+1,g}$ is fixed as
A stochastic first-order trust-region method with inexact…

\[ N_{k+1,g} = \lfloor cN_{k+1}^t \rfloor, \tag{68} \]

with \( c \in (0, 1) \) and \( I_{N_{k+1,g}} \subseteq I_{N_{k+1}}^t \).

### 4.1 SIRTR performance

In the following, we show the numerical behaviour of SIRTR on nonconvex binary classification problems. Let \( \{(a_i, b_i)\}_{i=1}^{N} \) denote the pairs forming a training set with \( a_i \in \mathbb{R}^n \) containing the entries of the \( i \)-th example, and \( b_i \in \{0, 1\} \) representing the corresponding label. Then, we address the following minimization problem

\[
\min_{x \in \mathbb{R}^n} f_N(x) = \frac{1}{N} \sum_{i=1}^{N} \left( b_i - \frac{1}{1 + e^{-a_i^T x}} \right)^2, \tag{69}
\]

where the nonconvex objective function \( f_N \) is obtained by composing a least-squares loss with the sigmoid function.

In Table 1, we report the information related to the datasets employed, including the number \( N \) of training examples, the dimension \( n \) of each example and the dimension \( N_T \) of the testing set \( I_N^t \).

We focus on three aspects: the classification error provided by the final iterate, the computational cost, the occurrence of termination before full accuracy in function evaluations is reached. The last issue is crucial because it indicates the ability of the inexact restoration approach to solve (69) with random models and to rule sampling and steplength selection.

The average classification error provided by the final iterate, say \( x_{\text{fin}} \), is defined as

| Data set | Training set | Testing set |
|----------|--------------|-------------|
|          | \( N \) | \( n \) | \( N_T \) |
| A8A [29] | 15887 | 123 | 6809 |
| A9A [29] | 22793 | 123 | 9768 |
| CINA0 | 10000 | 132 | 6033 |
| COD-RNA [30] | 41675 | 8 | 17860 |
| COVERTYPE [29] | 464810 | 54 | 116202 |
| HTRU2 [29] | 10000 | 8 | 7898 |
| ICNN1 [30] | 49990 | 22 | 91701 |
| MNIST [31] | 60000 | 784 | 10000 |
| PHISHING [30] | 7739 | 68 | 3316 |
| REAL-SIM [30] | 50616 | 20958 | 21693 |
| W7A [30] | 17284 | 300 | 7408 |
| W8A [30] | 34824 | 300 | 14925 |
where \( b_i \) is the exact label of the \( i \)-th instance of the testing set, and \( b_{i}^{\text{pred}} \) is the corresponding predicted label, given by \( b_{i}^{\text{pred}} = \max\{\text{sign}(a_{i}^{T}x_{\text{fin}}), 0]\).

The computational cost is measured in terms of full function and gradient evaluations. In our test problems, the main cost in the computation of \( /u1D719_{i} \), \( 1 \leq i \leq N \), is the scalar product \( a_{i}^{T}x \); once this product is evaluated, it can be reused for computing \( \nabla /u1D719_{i} \). Nonetheless, following [32, Section 3.3], we count both function and gradient evaluations as if we were addressing a classification problem based on a neural net. Thus, computing a single function \( \phi_{i} \) requires \( 1/N \) forward propagations, whereas the gradient evaluation corresponds to \( 2/N \) propagations (an additional backward propagation is needed). Note that, once \( \phi_{i} \) is computed, the corresponding gradient \( \nabla \phi_{i} \) requires only \( 1/N \) backward propagations. Hence, as in our implementation \( I_{N_k+1,g} \subseteq I_{N_k+1} \), the computational cost of SIRTR at each iteration \( k \) is determined by \( N_{k+1} \) propagations.

For all experiments in this section, we run SIRTR with \( x_0 = (0, 0, \ldots, 0)^T \) as initial guess, and stop it when either a maximum of 1000 iterations is reached or a maximum of 500 full function evaluations is performed or the condition

\[
|f_{N_k}(x_k) - f_{N_{k-1}}(x_{k-1})| \leq \epsilon |f_{N_{k-1}}(x_{k-1})| + \epsilon,
\]

with \( \epsilon = 10^{-3} \), holds for a number of consecutive successful iterations such that the computational effort is equal to the effort needed in three iterations with full function and gradient evaluations.

### Table 2

| \( N_{k+1,g} \) | [0.1/\( N_{k+1} \)] | [0.2/\( N_{k+1} \)] | \( N_{k+1} \) |
|---------------|----------------|----------------|-----------|
|               | Cost  | Err  | Sub | Cost  | Err  | Sub | Cost  | Err  | Sub |
| A8a           | 20    | 0.170| 15  | 19    | 0.171| 19  | 22     | 0.173| 29  |
| A9a           | 20    | 0.167| 12  | 17    | 0.169| 18  | 19     | 0.172| 13  |
| CINA0         | 72    | 0.146| 0   | 84    | 0.140| 0   | 116    | 0.158| 1   |
| COD-RNA       | 44    | 0.109| 0   | 42    | 0.106| 1   | 45     | 0.119| 0   |
| COVTYPE       | 22    | 0.425| 4   | 19    | 0.424| 8   | 20     | 0.435| 5   |
| HTRU2         | 30    | 0.024| 7   | 25    | 0.024| 13  | 32     | 0.024| 16  |
| IJCNN1        | 22    | 0.087| 0   | 20    | 0.088| 0   | 20     | 0.086| 0   |
| MNIST2        | 22    | 0.154| 10  | 25    | 0.151| 12  | 29     | 0.152| 18  |
| PHISHING      | 48    | 0.105| 0   | 43    | 0.108| 0   | 48     | 0.119| 0   |
| REAL-SIM      | 56    | 0.268| 0   | 56    | 0.270| 0   | 57     | 0.294| 0   |
| W7A           | 15    | 0.079| 22  | 15    | 0.079| 21  | 16     | 0.079| 34  |
| W8A           | 13    | 0.080| 25  | 13    | 0.080| 23  | 17     | 0.080| 28  |
Since the selection of sets $I_{N_t+1}$ and $I_{N_t+1,g}$ for computing $f_{N_t}(x_k)$ and $g_k$ is random, we perform 50 runs of SIRTR for each test problem. Results are reported in tables where the headings of the columns have the following meaning: cost is the overall number of full function and gradient evaluations averaged over the 50 runs, err is the classification error given in (70) averaged over the 50 runs, sub the number of runs where the method is stopped before reaching full accuracy in function evaluations.

In a first set of experiments, we investigate the choice of $N_{k+1,g}$ by varying the factor $c \in (0, 1]$ in (68). In particular, letting $\tilde{c} = 1.2$ in (66), $\mu = 100/N$ in (67) and $N_0 = [0.1N]$ as in [19], we test the values $c \in \{0.1, 0.2, 1\}$. The results obtained are reported in Table 2. We note that the classification error slightly varies with respect to the choice of $N_{k+1,g}$, and that selecting $N_{k+1,g}$ as a small fraction of $N_{t+1}$ is quite convenient from a computationally point of view. By contrast, the choice $N_{k+1,g} = N_{t+1}$ leads to the largest computational costs without providing a significant gain in accuracy. Besides the cost per iteration, equal to $2N_{t+1}/\mu$ in this latter case, we observe that full accuracy in function evaluations is reached very often especially for certain datasets, see e.g., cina0, cod-rna, covtype, icnn1, phishing, real-sim. Remarkably, the results in Table 2 highlight that random models compare favourably with respect to cost and classification errors.

Next, we show that SIRTR computational cost can be reduced by slowing down the growth rate of $N_{t+1}$. This task can be achieved controlling the growth of $\tilde{N}_{k+1}$ which affects $N_{t+1}$ by means of (67). Letting $c = 0.1$, $\mu = 100/N$ and $N_0 = [0.1N]$, we consider the choices $\tilde{c} \in \{1.05, 1.1, 1.2\}$ in (66). Average results are reported in Table 3. We can observe that the fastest growth rate for $\tilde{N}_{k+1}$ is generally more expensive than the other two choices, while the classification

| $\tilde{N}_{k+1}$ | $\min\{N, [1.05N_k]\}$ | $\min\{N, [1.1N_k]\}$ | $\min\{N, [1.2N_k]\}$ |
|------------------|----------------|----------------|----------------|
|                  | cost | err | sub  | cost | err | sub  | cost | err | sub  |
| A8A              | 27   | 0.170 | 49 | 18 | 0.170 | 44 | 18 | 0.171 | 16 |
| A9A              | 27   | 0.164 | 49 | 18 | 0.164 | 38 | 20 | 0.168 | 12 |
| cina0            | 35   | 0.167 | 44 | 44 | 0.163 | 13 | 68 | 0.151 | 0 |
| cod-rna          | 28   | 0.117 | 49 | 38 | 0.108 | 17 | 45 | 0.102 | 0 |
| covtype          | 12   | 0.396 | 50 | 13 | 0.392 | 48 | 20 | 0.423 | 7 |
| htru2            | 30   | 0.022 | 46 | 24 | 0.022 | 26 | 25 | 0.024 | 11 |
| icnn1            | 21   | 0.089 | 50 | 16 | 0.086 | 49 | 22 | 0.088 | 0 |
| mnist2           | 19   | 0.144 | 50 | 18 | 0.144 | 42 | 23 | 0.152 | 12 |
| phishing         | 28   | 0.117 | 50 | 30 | 0.110 | 23 | 46 | 0.103 | 0 |
| real-sim         | 36   | 0.254 | 50 | 65 | 0.272 | 0  | 57 | 0.267 | 0 |
| w7A              | 26   | 0.078 | 50 | 18 | 0.078 | 46 | 14 | 0.079 | 22 |
| w8A              | 20   | 0.079 | 50 | 14 | 0.080 | 46 | 13 | 0.080 | 26 |
error is similar for all the three choices. Moreover, significantly for $\tilde{c} = 1.05$ most runs stopped before reaching full function accuracy.

We now analyze three different values, $N_0 \in \{\lceil 0.001N \rceil, \lceil 0.01N \rceil, \lceil 0.1N \rceil \}$, for the initial sample size $N_0$. We apply SIRTR with $\tilde{c} = 1.05$ in (66), $\mu = 100/N$ in (67), and $c = 0.1$ in (68). Results are reported in Table 4. We can see that, reducing $N_0$, the number of full function/gradient evaluations can further reduce in some datasets, and that for $N_0 = \lceil 0.01N \rceil$ the average classification error compares well with the error when $N_0 = \lceil 0.1N \rceil$; for instance, the best results for most datasets are obtained by shrinking $N_0$ to 1% of the maximum sample size. We

| $N_0$   | [0.001N] |   | [0.01N] |   | [0.1N] |   |
|--------|---------|---|---------|---|--------|---|
|        | Cost    | Err | Sub     |   | Cost    | Err | Sub     |   | Cost    | Err | Sub     |   |
| A8A    | 30      | 0.182 | 50      |   | 30      | 0.169 | 47      |   | 28      | 0.170 | 50      |   |
| A9A    | 27      | 0.177 | 50      |   | 28      | 0.165 | 50      |   | 25      | 0.165 | 50      |   |
| CINA0  | 43      | 0.111 | 37      |   | 33      | 0.133 | 43      |   | 34      | 0.162 | 44      |   |
| COD-RNA| 4       | 0.412 | 50      |   | 25      | 0.194 | 50      |   | 29      | 0.114 | 48      |   |
| COVTYPE| 6       | 0.406 | 50      |   | 8       | 0.403 | 50      |   | 12      | 0.406 | 50      |   |
| HTRU2  | 38      | 0.036 | 40      |   | 35      | 0.021 | 43      |   | 31      | 0.021 | 47      |   |
| ICNN1  | 24      | 0.095 | 50      |   | 25      | 0.095 | 50      |   | 19      | 0.091 | 50      |   |
| MNIST2 | 18      | 0.185 | 50      |   | 20      | 0.160 | 50      |   | 21      | 0.143 | 50      |   |
| PHISHING| 4      | 0.410 | 50      |   | 28      | 0.163 | 48      |   | 29      | 0.118 | 50      |   |
| REAL-SIM| 4      | 0.188 | 50      |   | 5       | 0.166 | 50      |   | 35      | 0.254 | 50      |   |
| W7A    | 28      | 0.077 | 50      |   | 27      | 0.077 | 50      |   | 25      | 0.078 | 50      |   |
| W8A    | 23      | 0.078 | 50      |   | 23      | 0.079 | 50      |   | 20      | 0.079 | 50      |   |

| Table 4 Results with three different initial sample sizes $N_0$ |

| $N_0$   | [0.001N] |   | [0.01N] |   | [0.1N] |   |
|--------|---------|---|---------|---|--------|---|
|        | Cost    | Err | Sub     |   | Cost    | Err | Sub     |   | Cost    | Err | Sub     |   |
| A8A    | 30      | 0.182 | 50      |   | 30      | 0.169 | 47      |   | 28      | 0.170 | 50      |   |
| A9A    | 27      | 0.177 | 50      |   | 28      | 0.165 | 50      |   | 25      | 0.165 | 50      |   |
| CINA0  | 43      | 0.111 | 37      |   | 33      | 0.133 | 43      |   | 34      | 0.162 | 44      |   |
| COD-RNA| 4       | 0.412 | 50      |   | 25      | 0.194 | 50      |   | 29      | 0.114 | 48      |   |
| COVTYPE| 6       | 0.406 | 50      |   | 8       | 0.403 | 50      |   | 12      | 0.406 | 50      |   |
| HTRU2  | 38      | 0.036 | 40      |   | 35      | 0.021 | 43      |   | 31      | 0.021 | 47      |   |
| ICNN1  | 24      | 0.095 | 50      |   | 25      | 0.095 | 50      |   | 19      | 0.091 | 50      |   |
| MNIST2 | 18      | 0.185 | 50      |   | 20      | 0.160 | 50      |   | 21      | 0.143 | 50      |   |
| PHISHING| 4      | 0.410 | 50      |   | 28      | 0.163 | 48      |   | 29      | 0.118 | 50      |   |
| REAL-SIM| 4      | 0.188 | 50      |   | 5       | 0.166 | 50      |   | 35      | 0.254 | 50      |   |
| W7A    | 28      | 0.077 | 50      |   | 27      | 0.077 | 50      |   | 25      | 0.078 | 50      |   |
| W8A    | 23      | 0.078 | 50      |   | 23      | 0.079 | 50      |   | 20      | 0.079 | 50      |   |

Table 5 Average sample size $N_{fin}$ obtained at the final iteration, relative standard deviation $s$, and minimum and maximum sample sizes $N_{fin}^{min}, N_{fin}^{max}$ observed at the final iteration. Parameters setting: $N_0 = \lceil 0.01N \rceil$, $N_{k+1} = \lceil 0.1N_{k+1} \rceil$, $N_{k+1} = \min(N, \lceil 1.05N_k \rceil)$, $\mu = 100/N$

| $N_{fin}$ | $N_{fin}^{min}$ | $N_{fin}^{max}$ | $s$ | $N_{fin}^{min}$ | $N_{fin}^{max}$ | $s$ | $N_{fin}^{min}$ | $N_{fin}^{max}$ | $s$ |
|-----------|-----------------|-----------------|----|-----------------|-----------------|----|-----------------|-----------------|----|
| $N_0$     |                 |                 |    |                 |                 |    |                 |                 |    |
| A8A       | 15888           | 159             | 10353| 0.17           | 7407           | 13309|                |                 |    |
| A9A       | 22793           | 228             | 13637| 0.22           | 6718           | 18730|                |                 |    |
| CINA0     | 10000           | 100             | 7603 | 0.23           | 4771           | 10000|                |                 |    |
| COD-RNA   | 7739            | 78              | 3210 | 0.74           | 578            | 7054|                |                 |    |
| COVTYPE   | 464810          | 4649            | 54762| 0.32           | 33057          | 100341|              |                 |    |
| HTRU2     | 10000           | 100             | 7902 | 0.22           | 3923           | 10000|                |                 |    |
| ICNN1     | 49990           | 500             | 26966| 0.23           | 15408          | 43508|                |                 |    |
| MNIST2    | 60000           | 600             | 22928| 0.34           | 4383           | 45684|                |                 |    |
| PHISHING  | 7739            | 78              | 3926 | 0.63           | 578            | 7739|                |                 |    |
| REAL-SIM  | 50617           | 507             | 3721 | 0.034          | 3604           | 4174|                |                 |    |
| W7A       | 17285           | 173             | 10334| 0.23           | 5802           | 14674|                |                 |    |
| W8A       | 34825           | 349             | 17244| 0.19           | 9005           | 26360|                |                 |    |
conclude pointing out that most of the runs are performed without reaching full precision in function evaluation.

As a further confirmation of the efficiency of SIRTR, in Table 5 we report the sample sizes obtained on average at the stopping iteration of SIRTR with parameters setting $N_0 = [0.01N]$, $N_{k+1} = [0.1N_{k+1}']$, $N_{k+1} = \min\{N, [1.05N_k]\}$, $\mu = 100/N$. More specifically, for each dataset, we show the mean value $N_{\text{fin}}$ obtained by averaging the sample sizes $N_{\text{fin},i}$, $1 \leq i \leq 50$, used at the final iteration of SIRTR, the relative standard deviation $s = \frac{1}{N_{\text{fin}}} \sqrt{\frac{\sum_{i=1}^{50} (N_{\text{fin},i} - \bar{N}_{\text{fin}})^2}{50}}$ as a measure of dispersion of the final sample sizes with respect to the mean value, and the minimum and maximum sample sizes $N_{\text{fin}}^{\text{min}}, N_{\text{fin}}^{\text{max}}$ observed at the final iteration out of the 50 runs. From the reported values, we deduce that SIRTR terminates with a final sample size which is much smaller, on average, than the maximum sample size $N$.

Fig. 1 Dataset A9A. Samples sizes $N_{k+1}$ and $\tilde{N}_{k+1}$ versus iterations with $\mu = 100/N$ (left) and $\mu = 1$ (right), respectively, obtained with a single run of SIRTR. Classification errors: $\text{err} = 0.187$ with $\mu = 100/N$, $\text{err} = 0.174$ with $\mu = 1$

Fig. 2 Dataset MNIST. Samples sizes $N_{k+1}$ and $\tilde{N}_{k+1}$ versus iterations with $\mu = 100/N$ (left) and $\mu = 1$ (right), respectively, obtained with a single run of SIRTR. Classification errors: $\text{err} = 0.154$ with $\mu = 100/N$, $\text{err} = 0.167$ with $\mu = 1$
Finally, in Figs. 1, 2, we report the plots of the sample sizes $N^t_{k+1}$ and $\tilde{N}_{k+1}$ with respect to the number of iterations, obtained by running SIRTR on the A9A and MNIST datasets, respectively. In particular, we let either $\mu = 100/N$ or $\mu = 1$ in the update rule (67), $\tilde{c} = 1.05$ in (66), $c = 0.1$ in (68) and $N_0 = [0.1N]$. Note that a larger $\mu$ allows for the decreasing of both $N^t_{k+1}$ and $\tilde{N}_{k+1}$ in the first iterations, whereas a linear growth rate is imposed only in later iterations. This behaviour is due to the update condition (67), which naturally forces $N^t_{k+1}$ to coincide with $\tilde{N}_{k+1}$ when $\delta_k$ is sufficiently small. For both choices of $\mu$, we see that $N^t_{k+1}$ can grow slower than $\tilde{N}_{k+1}$ at some iterations, thus reducing the computational cost per iteration of SIRTR.

4.2 Comparison with TRish

In this section we compare the performance of SIRTR with the so-called Trust-Region-ish algorithm (TRish) recently proposed in [33]. TRish is a stochastic gradient method based on a trust-region methodology. Normalized steps are used in a dynamic manner whenever the norm of the stochastic gradient is within a prefixed interval. In particular, the $k$–th iteration of TRish is given by

$$x_{k+1} = x_k - \begin{cases} 
\gamma_{1,k} \alpha_k g_k, & \text{if } \|g_k\| \in [0, \frac{1}{\gamma_{1,k}}) \\
\alpha_k \frac{g_k}{\|g_k\|}, & \text{if } \|g_k\| \in [\frac{1}{\gamma_{1,k}}, \frac{1}{\gamma_{2,k}}] \\
\gamma_{2,k} \alpha_k g_k, & \text{if } \|g_k\| \in ([\frac{1}{\gamma_{2,k}}, \infty) 
\end{cases}$$

where $\alpha_k > 0$ is the steplength parameter, $0 < \gamma_{2,k} < \gamma_{1,k}$ are positive constants, and $g_k \in \mathbb{R}^n$ is a stochastic gradient estimate. This algorithm has proven to be particularly effective on binary classification and neural network training, especially if compared with the standard stochastic gradient algorithm [33, Section 4].

For our numerical tests, we implement TRish with subsampled gradients $g_k = \nabla f_S(x_k)$ defined in (5). The steplength is constant, $\alpha_k = \alpha$, $\forall k \geq 0$, and $\alpha$ is chosen in the set \{10^{-3}, 10^{-1}, \sqrt{10^{-1}}, 1, \sqrt{10}\}. Following the procedure in [33, Section 4], we use constant parameters $\gamma_{1,k} \equiv \gamma_1$, $\gamma_{2,k} \equiv \gamma_2$ and select $\gamma_1$, $\gamma_2$ as follows. First, Stochastic Gradient algorithm [4] is run with constant steplength equal to 1; second, the average norm $G$ of stochastic gradient estimates throughout the runs is computed; third $\gamma_1$, $\gamma_2$ are set as $\gamma_1 = \frac{4}{G}$, $\gamma_2 = \frac{1}{2G}$.

First, we compare TRish with SIRTR on the nonconvex optimization problem (69), using A9A, htru2, MNIST, and phishing as datasets (see Table 1). Based on the previous section, we equip SIRTR with $N_0 = [0.01N]$, $N_{k+1} = [0.1N^t_{k+1}]$, $\tilde{N}_{k+1} = \min\{N, [1.05N_k]\}$, $\mu = 100/N$. In TRish, the sample size $S$ of the
A stochastic first-order trust-region method with inexact...

stochastic gradient estimates is $\lceil 0.01N \rceil$, which corresponds to the first sample size used in SIRTR. We run each algorithm for ten epochs on the datasets a9a and htru2 using the null initial guess. We perform 10 runs to report results on average.

After tuning, the parameter setting for TRish was $\gamma_1 \approx 34.5805, \gamma_2 \approx 4.3226$ for a9a, $\gamma_1 \approx 57.9622, \gamma_2 \approx 7.2453$ for htru2, $\gamma_1 \approx 23.4376, \gamma_2 \approx 2.9297$ for mnist, and $\gamma_1 \approx 50.6409, \gamma_2 \approx 6.3301$ for phishing. In Fig. 3, we report the decrease of the (average) classification error, training loss $f_N$ and testing loss, $f_N T(x) = \frac{1}{N T} \sum_{i \in I_{N T}} \phi_i(x)$, over the (average) number of full function and gradient evaluations required by the algorithms. From these plots, we can see that SIRTR
performs comparably to the best implementations of TRish on a9a, htru2, mnist, while showing a good, though not optimal, performance on phishing.

In accordance to the experience in [33], all parameters $\gamma_1$ and $\gamma_2$ and $\alpha$ are problem-dependent. For instance, the best performance of TRish is obtained with $\alpha = 10^{-1}$ for a9a and with $\alpha = 10^{-3}$ for htru2, respectively; by contrast, SIRTR performs well with a unique setting of the parameters which is the key feature of adaptive stochastic optimization methods.

As a second test, we compare the performance of SIRTR and TRish on a different nonconvex optimization problem arising from nonlinear regression. Letting $\{(a_i, b_i)\}_{i=1}^N$ denote the training set, where $a_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$ represent the feature vector and the target variable of the $i$-th example, respectively, we aim at solving the following problem

$$\min_{x \in \mathbb{R}^n} f_n(x) = \frac{1}{N} \sum_{i=1}^N (b_i - h(a_i; x))^2,$$  \hspace{1cm} (72)

where $h(\cdot; x) : \mathbb{R}^n \to \mathbb{R}$ is a nonlinear prediction function.

For this second test, we use the air dataset [29], which contains 9358 instances of (hourly averaged) concentrations of polluting gases, as well as temperatures and relative/absolute air humidity levels, recorded at each hour in the period March 2004 - February 2005 from a device located in a polluted area within an Italian city.

As in [34], our goal is to predict the benzene (C6H6) concentration from the knowledge of $n = 7$ features, including carbon monoxide (CO), nitrogen oxides (NOx), ozone (O3), non-metanic hydrocarbons (NMHC), nitrogen dioxide (NO2), air temperature, and relative air humidity. First, we preprocess the dataset by removing examples for which the benzene concentration is missing, reducing the dataset dimension from 9357 to 8991. Then, we employ 70\% of the dataset for training ($N = 6294$), and the remaining 30\% for testing ($N_T = 2697$). Since the concentration values have been recorded hourly, this means that we use the data measured in the first 9 months for the training phase, and the data related to the

![Figure 4](image-url) Dataset air. Average testing loss (left) and training loss (right) versus epochs

© Springer
last 3 months for the testing phase. Finally, denoting with \( D = (d_{ij}) \in \mathbb{R}^{(N+N_T) \times n} \) the matrix containing all the dataset examples along its rows, and setting

\[
\begin{align*}
    m_j &= \min_{i=1,\ldots,N+N_T} d_{ij}, \\
    M_j &= \max_{i=1,\ldots,N+N_T} d_{ij}, \quad j = 1, \ldots, n,
\end{align*}
\]

we scale all data values into the interval \([0, 1]\) as follows

\[
d_{ij} = \frac{d_{ij} - m_j}{M_j - m_j}, \quad i = 1, \ldots, N+N_T, \quad j = 1, \ldots, n.
\]

We apply SIRTR and TRish on problem (72), where the prediction function \( h(\cdot; x) \) is chosen as a feed-forward neural network based on a \( 7 \times 5 \times 1 \) architecture (see [34] and references therein), with the two hidden layers both equipped with the linear activation function, and the output layer with the sigmoid activation function. We equip the two algorithms with the same parameter values employed in the previous tests, and run them 10 times for 10 epochs, using a random initial guess in the interval \([-\frac{1}{2}, \frac{1}{2}]\).

In Fig. 4, we report the decrease of the (average) training and testing losses provided by SIRTR and by TRish with different choices of the steplength \( \alpha \), whereas in Fig. 5 we show the benzene concentration estimations provided by the algorithms against the true concentration. These results confirm that the performances of SIRTR are comparable with those of TRish equipped with the best choice of the steplength and show the ability of SIRTR to automatically tune the steplength so as to obtain satisfactory results in terms of testing and training accuracy.
5 Conclusions

We proposed a stochastic gradient method coupled with a trust-region strategy and an inexact restoration approach for solving finite-sum minimization problems. Functions and gradients are subsampled and the batch size is governed by the inexact restoration approach and the trust-region acceptance rule. We showed the theoretical properties of the method and gave a worst-case complexity result on the expected number of iterations required to reach an approximate first-order optimality point. Numerical experience showed that the proposed method provides good results keeping the overall computational cost relatively low.

Funding Open access funding provided by Università degli Studi di Firenze within the CRUI-CARE Agreement. The research that led to the present paper was partially supported by a grant of the group GNCS of INdAM and partially developed within the Mobility Project: “Second order methods for optimization problems in Machine Learning” (ID:RS19MO05) executive programme of Scientific and Technological cooperation between the Italian Republic and the Republic of Serbia 2019-2022. The work of the second author was supported by Serbian Ministry of Education, Science and Technological Development, Grant No. 451-03-9/2021-14/200125.

Data availability The dataset CINA0 is no longer available in repositories but is available from the corresponding author on reasonable request. The other datasets analyzed during the current study are available in the repositories: http://www.csie.ntu.edu.tw/~cjlin/libsvm, http://yann.lecun.com/exdb/mnist, https://archive.ics.uci.edu/ml/index.php

Declarations

Conflict of interest All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Bellavia, S., Bianconcini, T., Krejić, N., Morini, B.: Subsampled first-order optimization methods with applications in imaging. In: Chen, K., Schönlieb, C., Tai, X., Younces, L. (eds.) Handbook of Mathematical Models and Algorithms in Computer Vision and Imaging. Springer, Switzerland AG (2021)
2. Curtis, F.E., Scheinberg, K.: Optimization methods for supervised machine learning: from linear models to deep learning. Leading Developments from INFORMS Communities. INFORMS, pp. 89–114 (2017)
A stochastic first-order trust-region method with inexact...
30. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2011) http://www.csie.ntu.edu.tw/cjlin/libsvm

31. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In Proceedings of the IEEE, vol. 86, pp. 2278–2324 (1998)

32. Xu, P., Roosta-Khorasani, F., Mahoney, M.W.: Second-order optimization for non-convex machine learning: an empirical study. In Proceedings of the 2020 SIAM International Conference on Data Mining, pp. 199–207 (2020)

33. Curtis, F.E., Scheinberg, K., Shi, R.: A stochastic trust region algorithm based on careful step normalization. INFORMS J. Optim. 1(3), 200–220 (2019)

34. De Vito, S., Massera, E., Piga, M., Martinotto, L., Di Francia, G.: On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. Sens. Actuator B 129, 750–757 (2008)

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.