STOCHASTIC TRUST-REGION METHODS WITH TRUST-REGION RADIUS DEPENDING ON PROBABILISTIC MODELS

Xiaoyu Wang\textsuperscript{1)}
Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China; University of Chinese Academy of Sciences, Beijing 100049, China
Email: wxy@lsec.cc.ac.cn
Ya-xiang Yuan
State Key Laboratory of Scientific/Engineering Computing, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China
Email: yyx@lsec.cc.ac.cn

Abstract

We present a stochastic trust-region model-based framework in which its radius is related to the probabilistic models. Especially, we propose a specific algorithm termed STRME, in which the trust-region radius depends linearly on the gradient used to define the latest model. The complexity results of the STRME method in nonconvex, convex and strongly convex settings are presented, which match those of the existing algorithms based on probabilistic properties. In addition, several numerical experiments are carried out to reveal the benefits of the proposed methods compared to the existing stochastic trust-region methods and other relevant stochastic gradient methods.

Mathematics subject classification: 65K05, 65K10, 90C60.
Key words: Trust-region methods, Stochastic optimization, Probabilistic models, Trust-region radius, Global convergence.

1. Introduction

In this paper, we are concerned with the following unconstrained optimization problem

$$\min_{x \in \mathbb{R}^d} f(x),$$

where the objective function $f$ is assumed to be smooth and bounded from below. But we only have access to the function value and its derivative information with some noise. In recent years, the expected risk minimization (ERM) problem, which is fundamental in the field of machine learning and statistics, has become the focus of many researchers. The ERM problems can be formulated as follows:

$$\min_{x \in \mathbb{R}^d} f(x) = \mathbb{E} [f(x; \xi)],$$

where $\mathbb{E} [\cdot]$ denotes the expectation taken with respect to the random noise variable $\xi$. However, because the probability distribution of $\xi$ is unknown in advance, solving (1.2) is intractable...
directly. Usually only noisy information about the gradient of $f$ is available. The empirical risk problem with a fixed amount of data (possibly very large) or the on-line setting problem where the data is flowing sequentially, which involves an estimate of problem (1.2), is more often considered in practice. Throughout the whole paper, we mainly consider stochastic optimization methods to solve such kind of problems.

1.1. Related work

The classic stochastic optimization method is stochastic gradient descent (SGD) method, which dates back to the work by [1]. The method is prominent in large-scale machine learning due to its simplicity and low-cost computing. However, because of the variance introduced by random sampling, the sequence of learning rate (step-size) progressively diminishes both in theoretical analysis and practical implementation, which leads to slow convergence. Thus finding an appropriate learning rate is critical for the performance of the SGD method, but it is not easy in practice. There are many existing methods to deal with aforementioned issues, and most of them can be classified into the following categories.

Adaptive stochastic gradient methods, e.g., AdaGrad [2], RMSProp [3], Adam [4], AMSGrad [5], have emerged to alleviate the burden to tune step size, which are widely used in deep learning.

Variance reduction methods have been proposed to improve the convergence of the SGD method, such as SVRG [6], SAGA [7] and SARAH [8]. Especially, they have achieved linear convergence rate when solving strongly convex problems, which is a stronger result than that of the SGD method. Furthermore, these methods have also been extended to solve nonconvex problems such as the deep neural networks [9–11]. The variance reduction technique is applicable to the problem with a large but fixed sample set, for which the full gradient has to be calculated as a compromise to achieve the significant variance reduction. Hence they are not easy to fit on the on-line setting like the SGD method and adaptive gradient methods.

Trust-region based algorithms. Recently, with the success of deep neural networks, the development and analysis of methods for nonconvex problems have attracted tremendous attention. As we know, the traditional trust-region method is a class of well-established and effective method in nonlinear optimization [12, 13]. An advantage of trust-region methods is that they can be efficient on nonconvex and ill-conditioned problems since they can make use of curvature information. Besides, due to the boundedness of the trust-region, the Hessian approximation matrix is not required to be positive definite. The trust-region based framework has already been considered to solve machine learning problems [14–20]. A saddle free Newton (SFN) method [16] is proposed to exploits the exact Hessian information to escape saddle points. However its computation is high cost for large-scale and high-dimension problems. A two-stage subspace trust-region approach [17] is proposed to train deep neural networks, in which the local second-order model is conducted based on the partial information computed from a subset of the data. But the approach lacks theoretical guarantees. Some literatures, for example, [14, 15], focus a specific class of machine learning problems, which need to utilize the accurate derivative information to construct good models, and the accurate function values to obtain good estimators. In [18], inexact Hessian information is incorporated into the trust-region framework but the exact gradient and function values are required to be computed per iteration. In [19] and [20], the authors construct the inexact models to satisfy some first-order accurate conditions with sufficiently high probability when building the trust-region subproblem. The complexity of line search and cubic regularization algorithms is analysed [21]
in nonconvex, convex and strongly convex settings, which is based on the random models with certain probability but their function evaluations are exact. The approach [21] is extended by [22] to the case of noisy function.

A stochastic trust-region algorithm named STORM for stochastic optimization setting has been introduced in [23]. Not only the model is conducted to satisfy some first-order accuracy requirements, but also the function values both at current iterate and next potential iterate are estimated with probability, instead of the exact function values. The liminf-type and lim-type first-order convergence have been analysed [23]. Besides, the expected convergence rates for both the first-order STORM and the second-order STORM algorithms are derived by [24]. Bellavia et.al [25] focuses on the stochastic complexity analysis of adaptive cubic regularization algorithm under inexact derivative information, in which inherits the adaptive accuracy requirements for Hessian approximations, while additionally employs inexact computations of the gradient.

1.2. Motivation and challenging

Previous works have established the convergence and complexity properties for the stochastic trust-region based framework. In this paper, we are particularly interested in introducing a new trust-region radius formula which depends on the latest probabilistic model, to improve the practical performance of such trust-region methods. Besides, we present an algorithm termed STRME in which the trust-region radius depends linearly on the gradient just computed, following a piece of work initially proposed for deterministic optimization [26]. The idea is meaningful and attractive. Because the trust region can be tailored according to the newly generated model, not just the success of the trial steps as that of STORM. Recently, Curtis & Scheinberg [27] also makes a short discussion about this type of trust region radius. A stochastic line search algorithm [28] is proposed, of which the gradient and function estimators are both randomly sampled with some probability. For the line-search type method, the search direction should be determined in advance. However, for the trust-region framework, we search a descent direction on the radius of the trust region. This is the fundamental difference between the two algorithms.

Note that the trust-region radius of STORM at iteration $k$ is completely determined by the past iterations from 0 to $k - 1$. However, the framework we proposed is no longer that case. That is to say the trust-region radius in our STRME algorithm, which related to the current model, is not measurable with respect to all the information generated from the past iterations from 0 to $k - 1$. It will bring new challenges both in theoretical analysis and numerical experiments. The trust-region radius in STRME depends on the newly updated model, which makes it more complicated to analyse the complexity results, compared to that of STORM. To obtain the complexity results of STRME, we analyse the parameter $\Lambda_k$ (called relative trust-region radius), which depends on the previous iterations, instead of the trust-region radius itself. The approach avoids the difficulty of analysing the trust-region radius directly. In [28], the authors also tackle this issue, of which the quantity $\Delta_k$ is related to the currently updated gradient. However, the quantity $\Delta_k$ in [28] is not really the trust-region radius. Our analysis is some what similar to that of [28]. However there are many distinctions, which will be discussed in details in Section 3. The convergence analysis for such trust-region algorithms relies on the requirements that these quantities such as models and function estimators are sufficiently accurate with sufficiently high probability. And the accuracy of these quantities are controlled by the trust-region radius. However the trust-region radius is unknown before the model is
updated. We will elaborate this at the beginning of Section 4 where we present numerical experiments.

1.3. Contributions

These changes in trust region radius indeed bring forth some advantages. The trust-region radius which depends on the model gradient can capture more new information. We have to say that it is advisable to adjust the trust-region radius according to the latest probabilistic model. If the stochastic trust-region algorithm is going to converge, then the inexact gradient with accuracy requirement approaches to the true gradient in certain probability. This means that the norm of the stochastic gradient somewhat reflects the training process. So our proposed algorithm can adaptively adjust the trust region radius. Besides, the trust region radius that normalized makes the algorithm keep relatively stable. Our numerical experiments illustrate this viewpoint. We have tested on regularized logistic regression and a simple deep neural network on real datasets. The numerical experiments show that the trust-region radius of the proposed STRME method reduces asymptotically. The oscillation in the trust-region radius is less severe in contrast to that of the STORM method. In addition, we observe that the proposed algorithm can get more successful iterates in a long time training session.

The major contributions of this paper are summarized as follows.

(1) The trust-region radius can be defined using the probabilistic model, in particular its gradient.

(2) We propose a specific algorithm termed STRME, in which the trust-region radius $\delta_k$ is linearly dependent on the norm of the model gradient. The complexity results of the STRME method in nonconvex, convex and strongly convex cases are presented, respectively. The expected number of iterations of the STRME algorithm for nonconvex problem is $O(\epsilon^{-2})$ by reaching $\|\nabla f(x)\| \leq \epsilon$ which is similar to the results of [24]. In addition, the expected convergence rates for general convex and strongly convex problems, are $O(1/\epsilon)$ and $O(\log(1/\epsilon))$ for reaching $f(x) - f^* \leq \epsilon$, respectively.

(3) In numerical experiments, sample averaging technique is utilized to construct probabilistic models. We adopt the dogleg method to approximately solve the trust-region subproblem for the regularized logistic regression problem. Moreover, the limited memory symmetric rank one (L-SR1) is employed to approximate the Hessian matrix, and then incorporated into STRME algorithm to train a deep neural network problem. The results indicate that the proposed algorithm compares favorably to other stochastic optimization algorithms.

The rest of this paper is organized as follows. In Section 2 we give some definitions about the probabilistic models and estimates; In Section 3 we propose a specific algorithm named STRME and analyse the complexity in nonconvex, convex and strongly convex cases; In Section 4, some numerical results on regularized logistic regression problem and a simple deep neural network are reported to show the efficiency of STRME in different settings; Finally, we conclude the paper in Section 5.

Notations. Throughout this paper, we use $x^* \in \mathbb{R}^d$ to denote the global minimizer, $f^* = f(x^*)$. Let $\|\cdot\|$ denote the Euclidean norm, i.e. $\|\cdot\|_2$, unless otherwise specified. Let $B(x, \Delta)$ denote the ball of the radius $\Delta$ around $x$. Let $\mathbb{I}\{A\}$ denote the indicator function of the event
A, i.e. if $A$ occurs, $\mathbb{I}\{A\} = 1$; else, $\mathbb{I}\{A\} = 0$. A function $f \in C^1(\mathbb{R}^d)$, if the first derivation of $f$ exists and continuous. A function $f$ is $L$-smooth, if there is a constant $L > 0$ such that

$$
\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|, \quad \forall x, y \in \mathbb{R}^d.
$$

2. Preliminaries on the Probabilistic Models and Estimates

Let us first introduce a generic stochastic trust-region framework. The analysis for the framework can particularize to the algorithm for example the algorithm STRME proposed in Section 3, and the objective function, whether it is nonconvex or convex, provided that the assumptions are satisfied.

**Algorithm 2.1. A Stochastic Trust Region Framework**

1: **Initialization**: Given an initial point $x_0$, $\gamma > 1$, $\eta_1 \in (0, 1)$, $\eta_2 > 0$, $\mu_0 \in (0, \mu_{\max})$ with $\mu_{\max} > 0$; Set $k = 0$
2: Construct a model (possibly random) $m_k(x_k + d)$ to approximate $f(x)$ at $x_k$ with $d = x - x_k$
3: Compute $\delta_k = \delta(m_k, \mu_k)$
4: Compute a trial step $d_k = \arg \min_{\|d\| \leq \delta_k} m_k(x_k + d)$ such that $d_k$ satisfies a sufficient reduction condition
5: Obtain estimates $f^0_k$ and $f^1_k$ of $f(x_k)$ and $f(x_k + d_k)$
6: Compute $\rho_k = \frac{f^0_k - f^1_k}{m_k(x_k + d_k) - m_k(x_k) - m_k(x_k + d_k)}$
7: if $\rho_k \geq \eta_1$ and $\|g_k\| \geq \eta_2 \delta_k$ then
8: $x_{k+1} = x_k + d_k$, $\mu_{k+1} = \min(\gamma \mu_k, \mu_{\max})$
9: else
10: $x_{k+1} = x_k$, $\mu_{k+1} = \mu_k / \gamma$
11: end if
12: Set $k := k + 1$, and go to step 2

The proposed Algorithm 2.1 covers the framework of STORM [23]. We can see that if $\delta_k = \mu_k$, the above algorithm will reduce to the STORM algorithm. The main difference lies in the trust-region radius in Algorithm 2.1 which depends on the current model $m_k$. We directly update the parameter $\mu_k$ (called relative trust-region radius), not the trust-region radius $\delta_k$ itself, which avoids the troubles by the randomness rise to the currently random model. Of course, except the current random model, there may be other factors, for instance the iterative models of previous steps, i.e. $\{m_t\}_{t \leq k}$, and the difference of previous iterates $\{x_t - x_{t-1}\}_{t \leq k}$ and so on. The framework we present here does not involve the specific forms of the random models and the sufficient reduction condition. The discussion will be presented in the next part.

Note that Algorithm 2.1 generates a random process. Obviously, the randomness of the algorithm comes from the randomness of the models and estimates we have constructed per iteration. At a deep level, it is determined by the inexact or random information obtained from the problem we are trying to solve. To formalize the random process, we introduce some notations to describe the quantities and their realizations. Let $M_k$ denote the random model in $k$-th iteration, while $m_k = M_k(\omega)$ for its realization, where $\omega$ is a random variable. We know
that the randomness of the models gives rise to the randomness of the iterates, relative trust-region radius and trial step produced by Algorithm 2.1. These random variables are denoted by \( X_k, \Lambda_k \) and \( D_k \), respectively, while let \( x_k = X_k(\omega), \mu_k = \Lambda_k(\omega) \), and \( d_k = D_k(\omega) \) to denote their realizations. Similarly, we use \( \{F^0_k, F^d_k\} \) to denote the random estimates of \( f(X_k) \) and \( f(X_k + D_k) \), while their realizations are denoted by \( f^0_k = F^0_k(\omega) \) and \( f^d_k = F^d_k(\omega) \). We will utilize those notations to analyse the random process later, which is under some assumptions that model \( M_k \) and estimates \( F^0_k, F^d_k \) are sufficiently accurate with some probability conditioned on the past. In order to formalize all the randomized information before \( k \)-th iteration, let \( \mathcal{F}^M_{k-1} \) denote the \( \sigma \)-algebra generated by \( \{M_0, \cdots, M_{k-1}\} \) and \( \{F^0_0, F^d_0, \cdots, F^0_{k-1}, F^d_{k-1}\} \). After the current model \( m_k \) is constructed, let \( \mathcal{F}^M_{k-1/2} \) denote the \( \sigma \)-algebra generated by \( \{M_0, \cdots, M_k\} \), and \( \{F^0_0, F^d_0, \cdots, F^0_{k-1}, F^d_{k-1}\} \).

Next, we will introduce some definitions to precise our requirements on the probabilistic models and estimates.

### 2.1. Probabilistic models and estimates

First we recall the measure for accuracy of deterministic models, which is introduced in [29,30].

**Definition 2.1.** Given a function \( f \in \mathcal{C}^1(\mathbb{R}^d) \) and constants \( \kappa_{ef}, \kappa_{eg} > 0 \). We say a model \( m_k \) is \( \kappa \)-fully linear model of function \( f \) on \( B(x_k, \delta_k) \), for \( \kappa = (\kappa_{ef}, \kappa_{eg}) \), if \( \forall y \in B(x_k, \delta_k) \),

\[
\|\nabla f(y) - \nabla m_k(y)\| \leq \kappa_{eg}\delta_k, \\
|f(y) - m_k(y)| \leq \kappa_{ef}\delta_k^2. \tag{2.1}
\]

The extending concept of the above definition is probabilistically fully-linear model, which is described in [23].

**Definition 2.2.** A sequence of random model \( \{M_k\} \) is said to be \( \alpha \)-probabilistically \( \kappa \)-fully linear with respect to the corresponding sequence \( \{X_k, \Lambda_k\} \), for each \( k \geq 1 \), if the events

\[
I_k = \mathbb{I}\{M_k \text{ is a } \kappa \text{-fully linear model of } f \text{ on } B(x_k, \delta(\mu_k))\}
\]

satisfy the condition

\[
\mathbb{P}(I_k = 1|\mathcal{F}^M_{k-1}) \geq \alpha, \tag{2.2}
\]

where \( \mathcal{F}^M_{k-1} \) is the \( \sigma \)-algebra generated by \( \{M_0, \cdots, M_{k-1}\} \) and \( \{F^0_0, F^d_0, \cdots, F^0_{k-1}, F^d_{k-1}\} \).

The above definition states that the model \( M_k \) is a locally good approximation of the first-order Taylor expansion of the objective function with probability at least \( \alpha \), conditioned on \( \mathcal{F}^M_{k-1} \). However, there is still some possibility such that the model is inaccurate, even very bad. To guarantee the quality of the trial step, we hope the random model \( M_k \) closer to the first-order Taylor expansion. However, the corresponding computation cost will increase. Thus there is a trade-off between the accuracy of the model and the computation cost.

Taking aside of the accurate model, the estimates of \( f(x_k) \) and \( f(x_k + d_k) \) are also required to be sufficiently accurate. The deterministic version of accurate estimates is formally stated as follows.
**Definition 2.3.** The estimates $f_k^0$ and $f_k^d$ are $\epsilon_F$-accurate estimates of $f(x_k)$ and $f(x_k + d_k)$, respectively, if

$$
\begin{align*}
|f_k^0 - f(x_k)| & \leq \epsilon_F \delta_k^2, \\
|f_k^d - f(x_k + d_k)| & \leq \epsilon_F \delta_k^2,
\end{align*}
$$

(2.3)

where $\epsilon_F > 0$ is a fixed constant.

The definition of probabilistically accurate estimates is shown as follows which is a modified version of that in [31].

**Definition 2.4.** A sequence of random estimates $\{F_k^0, F_k^d\}$ is said to be $\beta$-probabilistically $\epsilon_F$-accurate with respect to the corresponding sequence $\{X_k, \Lambda_k, D_k\}$, if the events

$$
J_k = 1 \{F_k^0 \text{ and } F_k^d \text{ are } \epsilon_F\text{-accurate estimates of } f(x_k) \text{ and } f(x_k + d_k) \text{, respectively}\}
$$

satisfy the condition

$$
\mathbb{P}(J_k = 1 | \mathcal{F}_{k-1/2}^{M,F}) \geq \beta,
$$

(2.4)

where $\mathcal{F}_{k-1/2}^{M,F}$ is the $\sigma$-algebra generated by $\{M_0, \cdots, M_k\}$ and $\{F_0^0, F_0^d, \cdots, F_{k-1}^0, F_{k-1}^d\}$.

Using Definitions 2.2 and 2.4, we assume that model $M_k$ and estimates $\{F_k^0, F_k^d\}$ satisfy the following assumption in our analysis.

**Assumption 2.1.** The followings hold for the quantities used in Algorithm 2.1

(i) There exist constants $\kappa_{ef}, \kappa_{eg} > 0$ such that the sequence of random models $M_k$ is $\alpha$-probabilistically $(\kappa_{ef}, \kappa_{eg})$-fully linear, for a sufficiently large $\alpha \in (0, 1)$.

(ii) There exists a constant $\epsilon_F > 0$ such that the sequence of estimates $\{F_k^0, F_k^d\}$ is $\beta$-probabilistically $\epsilon_F$-accurate, for a sufficiently large $\beta \in (0, 1)$.

(iii) The sequence of estimates $\{F_k^0, F_k^d\}$ generated by Algorithm 2.1 satisfies the following condition that

$$
\begin{align*}
\mathbb{E} \left[ |f_k^0 - f(X_k)| | \mathcal{F}_{k-1/2}^{M,F} \right] & \leq \kappa_f \Delta_k^2, \\
\mathbb{E} \left[ |f_k^d - f(X_k + D_k)| | \mathcal{F}_{k-1/2}^{M,F} \right] & \leq \kappa_f \Delta_k^2,
\end{align*}
$$

(2.5)

with $\kappa_f > 0$.

**Remark 2.1.** The above assumption is significant in the following analysis. Compared to the Assumption 3 in [24], we know that Assumption 2.1(iii) is additional. Nevertheless it is essential in our analysis, which states that estimates $\{F_k^0, F_k^d\}$ can not be too worse in expectation in contrast to the true function values $\{f(X_k), f(X_k + D_k)\}$, especially when model $M_k$ and estimates $\{F_k^0, F_k^d\}$ are not sufficiently accurate (see the case (a) of (iii) in the proof of Theorem 3.1 in Appendix B).

One may doubt that the above condition (2.5) is a little stronger than (2.3) of stochastic line search [28] shown as follows

$$
\begin{align*}
\mathbb{E} \left[ |f_k^0 - f(X_k)|^2 | \mathcal{F}_{k-1/2}^{M,F} \right] & \leq \max \left\{ \kappa_f^2 \Delta_k^2 \| \nabla f(X_k) \|^4, \theta^2 \Delta_k^4 \right\}, \\
\mathbb{E} \left[ |f_k^d - f(X_k + D_k)|^2 | \mathcal{F}_{k-1/2}^{M,F} \right] & \leq \max \left\{ \kappa_f^2 \Delta_k^2 \| \nabla f(X_k) \|^4, \theta^2 \Delta_k^4 \right\},
\end{align*}
$$

(2.6)
We would agree that (2.5) of Assumption 2.1 and (2.6) [28] are close in spirit. However, by carefully deduction, we find that it does not possibly obtain one from the another and vice-versa. In practice, $\Lambda_k \|\nabla f(X_k)\|^2$ is hard to be calculated directly when constructing the estimates $\{F_0^k, F_d^k\}$ to satisfy (2.6). However, our Assumption 2.1(iii) can easily be satisfied in the practical implementation as long as Assumption 2.1(ii) holds. We will explain this in our numerical experiments.

To make the analysis simple and easy to understand, we use the following statements.

- If $I_k = 1$, we say that the model is true; otherwise, we say that the model is false.
- If $J_k = 1$, we say that the estimates are tight; otherwise, we say that the estimates are loose.
- If an iteration $k$ is accepted, we say that the iteration is successful; otherwise, we say that the iteration is failed.

The analysis for the renewal-reward process in [24] is appropriate for the stochastic process $\{X_k, \Lambda_k\}$ generated by Algorithm 2.1, in which we uses $\Lambda_k$ as opposite to $\Delta_k$ in [24]. To highlight the focus of our work, we omit the analysis for the random process $\{X_k, \Lambda_k\}$ here. The details and discussions are attached in Appendix A. Different choices of $\Delta_k$ actually produce different stochastic processes $\{X_k, \Lambda_k\}$. In the next part, we will give a specific formula of the trust region radius $\Delta_k$ and elaborate its complexity results.

## 3. Stochastic Trust-Region with Probabilistic Models and Estimates

In this section, we propose a trust-region framework named STRME based on probabilistic models and estimates. The main steps are described as follows.

At each iteration $k$, given a current point $x_k$ and trust-region radius $\delta_k$, the quadratic model is built as

$$m_k(x_k + d) = f_k + g_k^T d + \frac{1}{2} d^T B_k d,$$

(3.1)

To approximate $f(x)$ in $B(x_k, \delta_k)$. The quadratic model is simple and widely used in many trust-region algorithms. Of course, other models, for example the conic model (see the survey by [13]), can also be applied to the framework as long as some requirements we stated are met.

The trust-region radius is defined as

$$\delta_k = \mu_k \|g_k\|,$$

where $\mu_k \in (0, \mu_{\text{max}}]$. Actually, one can try more general choices that $\delta_k = \mu_k^{r_1} \|g_k\|^{r_2}$ with $r_1, r_2 \geq 0$ [32]. For simplicity, we only consider the case that $r_1, r_2 = 1$. In the following steps, we choose to update the parameter $\mu_k$ (relative trust-region radius). The trial step $d_k$ is produced by minimizing the model $m_k(x_k + d)$ in a neighborhood of $x_k$ exactly or inexactly. Then we compute the random estimates $f_0^k$ and $f_d^k$ of $f(x_k)$ and $f(x_k + d_k)$, respectively, to measure the actual function reduction. One simple way is that we can use sample averaging technique [33] to construct stochastic gradient $g_k$, matrix $B_k$ and function estimates $(f_0^k, f_d^k)$. Note that due to the randomness of the model, $\|g_k\|$ can be very small even zero even though the algorithm has not converged. In order to cover this extreme case, we do not compute $\rho_k$ directly as Algorithm 2.1 and we use

$$f_0^k - f_d^k \geq \eta_1 (m_k(x_k) - m_k(x_k + d_k))$$
instead to judge how good the trial step $d_k$ is. Based on this criterion, if the trial point $x_k + d_k$ yields sufficient reduction, we accept the trial step $d_k$; otherwise, we reject it. At the end of each iteration, the relative trust-region $\mu_k$ is chosen according to the outcome of the iterates. The details of the algorithm are described as follows.

**Algorithm 3.1.** Stochastic Trust-Region with Probabilistic Model and Estimates (STRME)

1: **Initialization:** Given an initial point $x_0$, $\gamma > 1$, $\eta_1 \in (0, 1)$, $\mu_0 \in (0, \mu_{\max})$ with $\mu_{\max} > 0$; Set $k = 0$
2: Construct a (random) model $m_k(x_k + d) = f_k + g_k^T d + \frac{1}{2} d^T B_k d$ that approximates $f(x)$ at $x_k$ with $d = x - x_k$
3: Compute $\delta_k = \mu_k \|g_k\|$  
4: Compute the trial step $d_k = \arg\min_{\|d\| \leq \delta_k} m_k(x_k + d)$ (approximately) such that $d_k$ satisfies Assumption 3.1
5: Obtain estimates $f_k^0$ and $f_k^d$ of $f(x_k)$ and $f(x_k + d_k)$
6: if $f_k^0 - f_k^d \geq \eta_1 (m_k(x_k) - m_k(x_k + d_k))$ then
7: $x_{k+1} = x_k + d_k$, $\mu_{k+1} = \min(\gamma \mu_k, \mu_{\max})$
8: else
9: $x_{k+1} = x_k$, $\mu_{k+1} = \mu_k / \gamma$
10: end if
11: Set $k := k + 1$, and go to step 2

**Assumption 3.1.** For each $k \geq 1$, the step $d_k$ is computed to satisfy that

$$m_k(x_k) - m_k(x_k + d_k) \geq \kappa_{fcd} \|g_k\| \min \left\{ \frac{\|g_k\|}{\|B_k\|}, \delta_k \right\},$$

(3.2)

where $\kappa_{fcd} \in (0, 1)$.

At each iteration, the trial step $d_k$ is computed to satisfy Assumption 3.1 which is the well-known *Cauchy decrease* condition. As we know if $d_k$ is a Cauchy step, then (3.2) can be satisfied. Actually, the reduction on the model $m_k$ achieved by the exact solution will be not less than that of the Cauchy step. Thus, if $d_k$ is the exact solution of subproblem, we know it will satisfy Assumption 3.1.

Besides, in the case that $\delta_k = \mu_k \|g_k\|$, the condition $\|g_k\| \geq \eta_2 \delta_k$ in Algorithm 2.1 is equivalent to the condition $\mu_k \leq 1/\eta_2$. The parameter $\eta_2$ of STORM [23] is lower bounded by a positive number for theoretical analysis. In our analysis $\mu_k$ is required to be upper bounded by $\mu_{\max}$. However, there is no any constraint on $\mu_{\max}$.

**3.1. Theoretical properties of STRME**

We are ready to present the theoretical properties of the framework described in Algorithm 3.1. Firstly, we give an assumption which states that the Hessian approximation matrix $B_k$ in model $m_k$ is uniformly upper bounded.

**Assumption 3.2.** There exists a constant $\kappa_{hbm} > 0$ such that, for all $k \geq 0$,

$$\|B_k\| \leq \kappa_{hbm}.$$
We now provide some auxiliary lemmas to show that the decrease of the objective function $f(x)$ is guaranteed under some conditions. The proofs of the lemmas and theorems in this section are given in Appendix B.

**Lemma 3.1.** Suppose that model $m_k$ is true and Assumption 3.2 holds. If

$$
\mu_k \leq \min \left\{ \frac{\kappa_{fcd}}{8\kappa_{ef}}, \sqrt{\frac{\kappa_{fcd}}{8\kappa_{ef}\kappa_{bhm}}} \right\},
$$

then

$$
f(x_k) - f(x_k + d_k) \geq \frac{\kappa_{fcd}}{4} \min \left\{ \frac{1}{\kappa_{bhm}}, \mu_k \right\} \|g_k\|^2.
$$

Lemma 3.1 states that when model $m_k$ is true (fully linear), if the relative trust-region radius $\mu_k$ is upper bounded by a given number, no matter whether the iteration $k$ is accept or not, then the actual reduction of the objective function is achieved. If $\kappa_{fcd}/(8\kappa_{ef}) \leq 1/\kappa_{bhm}$, the result is an extension of Lemma 3 in [24], by taking $\delta_k = \mu_k \|g_k\|$. Otherwise, if $\kappa_{fcd}/(8\kappa_{ef}) > 1/\kappa_{bhm}$, we have $\mu_k \leq \sqrt{\kappa_{fcd}/(8\kappa_{ef}\kappa_{bhm})} > 1/\kappa_{bhm}$. In this case, we can see that $\mu_k$ is possibly larger than $1/\kappa_{bhm}$.

The next lemma states that the decrease of the objective function is achieved if the estimates $\{f_0^k, f_d^k\}$ are tight and the iteration $k$ is successful.

**Lemma 3.2.** Suppose that Assumption 3.2 holds, and estimates $f_0^k$ and $f_d^k$ are tight with $\epsilon_F \leq \frac{\kappa_{fcd}}{8\mu_{\text{max}}} \min \left\{ \frac{1}{\kappa_{bhm}}, 1 \right\}$.

If $d_k$ is accepted, then

$$
f(x_k) - f(x_k + d_k) \geq \frac{\eta \kappa_{fcd}}{4} \min \left\{ \frac{1}{\kappa_{bhm}}, \mu_k \right\} \|g_k\|^2.
$$

**Remark 3.1.** From Lemmas 3.1 and 3.2, we can see that whether model $m_k$ is true ($I_k = 1$) or estimates $f_0^k$ and $f_d^k$ are tight ($J_k = 1$), the decrease of the objective function is achieved under some conditions. Moreover, when $m_k$ is true, the decrease achieved in the objective function is larger than that of the tight estimates $f_0^k$ and $f_d^k$. It is natural to use the result of Lemma 3.1 to guarantee the better reduction in the objective function, when $I_k = 1$ and $J_k = 1$ (see the case (a) of (i) in the proof of Theorem 3.1). However, from Lemma 3.1, to obtain the decrease of the objective function, we require that $\mu_k$ is upper bounded by a given number, which is likely to be very small. This basically contradicts our idea that adapts the parameter $\mu_k$ per iteration. Thus, we settle for Lemma 3.2 in the proof of Theorem 3.1.

The following lemma shows when model $m_k$ and estimates $\{f_0^k, f_d^k\}$ are both sufficiently accurate, if $\mu_k$ is not too large, then the iteration will be successful.

**Lemma 3.3.** Suppose that Assumption 3.2 holds, model $m_k$ is true and estimates $f_0^k$ and $f_d^k$ are tight with $\epsilon_F \leq \kappa_{ef}$. If

$$
\mu_k \leq \min \left\{ \frac{\kappa_{fcd}(1 - \eta_1)}{8\kappa_{ef}}, \sqrt{\frac{\kappa_{fcd}(1 - \eta_1)}{8\kappa_{ef}\kappa_{bhm}}} \right\},
$$

then the $k$-th iteration is successful.
We now turn to consider the random process \{\Lambda_k, \Phi_k\} derived from the process generated from Algorithm 3.1. The following analysis is based on the function

\[ \Phi_k = \nu(f(X_k) - f^*) + (1 - \nu) \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2, \]  

(3.3)

for some \( \nu \in (0, 1) \). It is obvious that \( \Phi_k \geq 0 \). Actually, the random variable \( \Phi_k \) can be regarded as a kind of measure of progress to optimality. It plays an important role in the analysis of such trust-region algorithms. In Algorithm 3.1, we update the trust-region radius as \( \Delta_k = \Lambda_k \|G_k\| \). One may find that the trust-region radius \( \Delta_k \) depends on the randomness introduced by the current model \( M_k \), that is, \( \Delta_k \) is not measurable with respect to \( F_{k-1}^{M,F} \). We use the parameter \( \Lambda_k \), which is completely determined by \( F_{k-1}^{M,F} \), instead of \( \Delta_k \) as [23] to analyse the complexity of STRME. Our approach avoids the difficulty of using the trust-region radius directly to analyse the complexity of STRME.

Even though the analysis process above is similar to that of stochastic line search method [28], there are some differences between them. One difference lies in the construction of the measure function \( \Phi_k \). Due to the possibility of inaccurate model \( M_k \) and estimates \( \{ F_k^0, F_k^d \} \), the function value of \( f \) may increase. Thus, \( \Phi_k \) is designed to balance the decrease and increase of \( f(X_k) \). The decrease of expected \( \Phi_k \) can be achieved by carefully choosing \( \nu \) and the probability \( \alpha \) and \( \beta \), as shown above.

**Theorem 3.1.** We assume that \( f \) is \( L \)-smooth and Assumption 3.2 is satisfied. Let Assumptions 2.1 hold with

\[ \epsilon_F \leq \frac{\eta f_{cd}}{8\mu_{\max}} \min \left\{ \frac{1}{\kappa \beta h \mu_{\max}}, 1 \right\} \quad \text{and} \quad \kappa_f \leq \frac{\eta f_{cd}}{4\mu_{\max}} \min \left\{ \frac{1}{\kappa \beta h \mu_{\max}}, 1 \right\}. \]

Then there exist a constant \( \nu \) and sufficiently large \( \alpha, \beta \in (0, 1) \) satisfying the following conditions

\[ \frac{1 - \nu}{\nu} \leq \min \left\{ \frac{\kappa_f \eta_f}{16\gamma \mu_{\max}^2}, \frac{\kappa_f \eta_f L^2}{32\gamma (1 + \kappa_{eg} \mu_{\max})^2} \right\} \cdot \min \left\{ \frac{1}{\kappa \beta h \mu_{\max}}, 1 \right\}, \]

\[ \alpha \beta \geq \frac{4\gamma^2}{4\gamma^2 + (\gamma - 1)}, \]

such that

\[ \mathbb{E} \left[ \Phi_{k+1} - \Phi_k | F_{k-1}^{M,F} \right] \leq -\frac{1}{2} \alpha \beta (1 - \nu) (1 - \frac{1}{\gamma}) \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2. \]

(3.4)

Another obvious difference lies in the requirement on \( \alpha \) and \( \beta \), compared to the analysis of [28]. First of all, from analysis of Theorem 3.1, if Assumption 2.1(iii) is relaxed to assumption 2.4 [28], as shown that

\[ \max \left\{ \kappa_f \Lambda_k \|\nabla f(X_k)\|^2, \theta \Delta_k^2 \right\} \leq \kappa_f \Lambda_k \|\nabla f(X_k)\|^2 + \theta \Delta_k^2, \]

we see that the requirement for \( \alpha \) and \( \beta \) in [28] will be stronger than the current results. Furthermore, from Theorem 3.1, if \( \gamma = 2 \), we require that \( \alpha \beta \geq 16/17 \). However, from (4.32) and (4.33) in Theorem 4.7 of [28], we can see that not only \( \alpha \geq 16/17 \), but also

\[ \alpha \beta / \sqrt{1 - \beta} \geq \max \{1024 \kappa f L^2 (\kappa_{eg} \mu_{\max} + 1)^2, 1024 (\kappa_{eg} \mu_{\max} + 1)^2 \}. \]
Indeed, we have to admit that our condition \( \alpha \beta \geq \frac{16}{17} \) is stronger than \( \alpha \geq \frac{16}{17} \). However, we know that \( 0 < \alpha, \beta < 1 \), so \( \alpha \beta < 1 \). In order to satisfy the condition related to \( \alpha \beta / \sqrt{1 - \beta} \) [28], they have to require that

\[
1024(\kappa_{eg} \mu_{\text{max}} + 1)^2 \sqrt{1 - \beta} \leq 1.
\]

Then

\[
\beta \geq 1 - \frac{1}{1024^2(\kappa_{eg} \mu_{\text{max}} + 1)^2}.
\] (3.5)

For \( \kappa_{eg} \mu_{\text{max}} > 0 \), we know the right term of (3.5) is very close to 1. The condition in [28] related to \( \beta \) is stronger than ours. When \( \beta \) is close to the number one, the condition \( \alpha \beta \geq \frac{16}{17} \) will reduce to \( \alpha \geq \frac{16}{17} \). Thus we can conclude that our requirement on \( \alpha \) and \( \beta \) is looser than that of [28].

**Remark 3.2.** We use the additional Assumption 2.1(iii), which is closed to assumption 2.1 in [28], to control the case of loose estimates. In Theorem 3.1, we require an upper bound of \( \kappa_f \), i.e., \( \kappa_f \leq \frac{\eta_{f,\text{cd}}}{\kappa_{\text{eg}} \mu_{\text{max}}} \). Paquette and Scheinberg [28] do not require such upper bound for \( \kappa_f \), however they in turn add an extra condition for probabilities \( \alpha \) and \( \beta \) as discussed above.

Next, we show that such a requirement on \( \kappa_f \) is reasonable as a trade-off. In Theorem 3.1, we require that \( \epsilon_F \leq \frac{\eta_{f,\text{cd}}}{\kappa_{\text{eg}} \mu_{\text{max}}} \). If we choose the upper bound of \( \epsilon_F \), then \( \kappa_f \leq 2\epsilon_F \). We can see that

\[
\mathbb{E} \left| F_0^0 - F(X_k) \right| \leq 2\epsilon_F \Delta_k^2.
\]

Because \( \alpha \beta \geq \frac{4_{\text{eg}}^2}{4_{\text{eg}}^2 + (1 - 1)} \) (\( \alpha, \beta \in (0, 1) \)), we have that \( \beta \geq \frac{4_{\text{eg}}^2}{4_{\text{eg}}^2 + (1 - 1)} \). We can see that the loose estimate has an reasonable upper bound that \( |F_0^0 - F(X_k)| \leq (\frac{4_{\text{eg}}^2}{4_{\text{eg}}^2} + 2)\epsilon_F \Delta_k^2 \). Similarly, it is also applied to \( |F_0^d - F(X_k + D_k)| \).

### 3.2. Convergence rates for nonconvex problems

We now show the global convergence rate of Algorithm 3.1 when \( f \) is unknown to be convex, that is the following assumption holds.

**Assumption 3.3.** \( f \in \mathcal{C}^1(\mathbb{R}^d) \) is bounded below by \( f^* \) and \( L \)-smooth.

Our goal is to bound the expected number of iterations until an \( \epsilon \)-solution occurs, i.e. \( \mathbb{E} \left[ T_\epsilon \right] \). The definition of \( T_\epsilon \) is described as follows

\[
T_\epsilon = \inf \{ k \geq 0 : \| \nabla f(X_k) \| \leq \epsilon \}.
\] (3.6)

For the general case which is possibly nonconvex, we achieve the following complexity result. The details of the proof are provided in Appendix B.
Theorem 3.2. Suppose that $f$ satisfies Assumption 3.3. Under the conditions in Theorem 3.1, if $\alpha \beta > \frac{1}{2}$, then for Algorithm 3.1, to achieve $\epsilon$-solution, the expected number of iterations is bounded as follows
\[
\mathbb{E}[T_\epsilon] \leq \frac{\alpha \beta}{(2\alpha \beta - 1)} \cdot \frac{M}{\epsilon^2} + 1,
\] (3.7)
where $M = \frac{2\nu L^2 (f(x_0) - f^*)^2 + 2(1-\nu)\mu_0 \mu_0 \sigma^2}{\alpha \beta (1-\nu)(1-\frac{1}{\gamma})\hat{\Lambda}}$ and $\hat{\Lambda}$ is chosen as (B.10).

Remark 3.3. Note that the dependency of the expected number of iterations for obtaining an $\epsilon$-solution is $O(1/\epsilon^2)$, which is similar to those in [24] and [28].

3.3. Convergence rates for convex problems

In this part, we will analyse the expected complexity for STRME when $f$ is convex. First we give the following assumption.

Assumption 3.4. $f \in C^1(\mathbb{R}^d)$ is convex. The level set $\mathcal{L} = \{x \in \mathbb{R}^d : f(x) \leq f(x_0)\}$ is bounded, and there exists a constant $D > 0$ such that
\[
\|x - x^*\| \leq D, \quad \forall x \in \mathcal{L}.
\] (3.8)

In the convex setting, we aim to bound the expected $T_\epsilon$, which is defined as below
\[
T_\epsilon = \inf \{k \geq 0 : f(X_k) - f^* \leq \epsilon\},
\] (3.9)
for an $\epsilon$-solution. In this case, we define a function
\[
\Psi_k = \frac{1}{\nu \epsilon} - \frac{1}{\Phi_k},
\] (3.10)
for $k \leq T_\epsilon$, to replace $\Phi_k$ to measure the progress of the iterations. For $k \leq T_\epsilon$, $f(X_k) - f^* \geq \epsilon$, then we have $\Phi_k \geq \nu \epsilon$. Thus $\Psi_k \geq 0$, which is well-defined. Next we give the complexity result for general convex problems as follows.

Theorem 3.3. We assume that $f$ is $L$-smooth and satisfies Assumption 3.4. Under the conditions in Theorem 3.1 and $\alpha \beta > \frac{1}{2}$, for Algorithm 3.1, in order to achieve an $\epsilon$-solution, we have
\[
\mathbb{E}[T_\epsilon] \leq \frac{\alpha \beta}{(2\alpha \beta - 1)} \cdot \frac{M}{\epsilon} + 1,
\] (3.11)
where $M = \frac{2(\nu L^2(1-\nu)\mu_{\text{max}})^2 D^2}{\alpha \beta (1-\nu)(1-\frac{1}{\gamma})\hat{\Lambda}}$.  

Remark 3.4. The above theorem states that if $f$ is convex, the dependency on $\epsilon$ for the complexity bound is $O(1/\epsilon^2)$, which is a stronger result than that in Theorem 3.2. Note that the above result is the same as that of traditional trust-region for general convex problems and is comparable to that of the stochastic line search method [28].
3.4. Convergence rates for strongly convex problems

In this part, we will derive the complexity bound for Algorithm 3.1 in the strongly convex setting. In this case, we assume the following assumption holds.

**Assumption 3.5.** \( f \in C^1(\mathbb{R}^d) \) is strongly convex, i.e. there exists a constant \( \sigma > 0 \) such that

\[
f(x) \geq f(y) + \nabla f(y)^T (x - y) + \frac{\sigma}{2} \|x - y\|^2, \quad \forall x, y \in \mathbb{R}^d.
\] (3.12)

Firstly, we reuse the definition of \( T_\epsilon \) same as (3.9). Our aim is to bound the expected number of iteration i.e. \( \mathbb{E}[T_\epsilon] \), to obtain an \( \epsilon \)-solution for strongly convex problems. Instead of using \( \Phi_k \) to measure the progress towards optimality, we define a function

\[
\Psi_k = \log(\Phi_k) + \log \left( \frac{1}{\nu \epsilon} \right), \quad k \leq T_\epsilon.
\] (3.13)

For \( k \leq T_\epsilon \), \( f(X_k) - f^* \geq \epsilon \), then we have \( \Phi_k \geq \nu \epsilon \), which implies that \( \Psi_k \geq 0 \). So the definition of \( \Psi_k \) is reasonable. In the later part, we will obtain the upper bound of \( \mathbb{E}[T_\epsilon] \) with the help of random process \( \{A_k, \Psi_k\} \). The details will be shown in the proof of the following theorem.

**Theorem 3.4.** Assume that \( f \) is \( L \)-smooth and satisfies Assumption 3.5. If the conditions in Theorem 3.1 and \( \alpha \beta > \frac{1}{2} \) hold, then for Algorithm 3.1, in order to achieve an \( \epsilon \)-solution, we have

\[
\mathbb{E}[T_\epsilon] \leq \frac{\alpha \beta}{2\alpha \beta - 1} \cdot M \log \left( \frac{1}{\epsilon} \right) + 1,
\] (3.14)

where

\[
M = \left( \frac{(\nu^2 + (1-\nu)\mu_{\text{max}})}{\alpha \beta (1-\nu) (1-\frac{1}{\Lambda})} \right) - 1.
\]

**Remark 3.5.** The result in Theorem 3.4 shows that Algorithm 3.1 takes at most \( O(\log(1/\epsilon)) \) iterations in expectation, to achieve an \( \epsilon \)-solution. The result coincides with that for trust-region methods in deterministic setting and is similar to that of [28].

4. Numerical Experiments

In this section, we empirically test our STRME algorithm and compare its performance with STORM [23] and some related algorithms.

We test on two type of problems: (i) regularized logistic regression problem, which is strongly convex; (ii) deep neural networks, which is highly nonlinear and nonconvex. The function value on training data called training loss (or its error i.e., \( f(x_k) - f^* \) called training loss error)) and accuracy (percentage of correctly classified testing data) are adopted as criteria to measure the performance of all the algorithms that are tested. For all those algorithms, we compare these criteria against the number of effective pass through the data, that is total gradient calls (denoted by \# grad) divided by \( n \) (training data size). All algorithms were terminated when the maximum budget of the gradient evaluations is larger than the maximum value \( SFO_{\text{max}} \).

All algorithms are implemented in Anaconda3 (Python 3.6.2) under Windows 7 operating system on Dell desktop with Intel(R) Core(TM) i7-4790U CPU @3.6GHz, 8GB Memory.
4.1. How to implement the probabilistic models and estimates to satisfy Assumption 2.1

In these numerical experiments, we focus on the derivative-based problems where \( f(x, \xi) \) and \( \nabla f(x, \xi) \) are available. At each iteration point \( X_k \), we assume that the noise \( \xi \) in function and gradient computation is conditionally unbiased and the corresponding variance is conditionally bounded for all \( f \), i.e.

\[
\begin{align*}
\mathbb{E} \left[ f(X_k; \xi) \mid \mathcal{F}_{k-1}^{M,F} \right] &= f(X_k), \\
\text{Var} \left[ f(X_k; \xi) \mid \mathcal{F}_{k-1}^{M,F} \right] &\leq V_f; \\
\mathbb{E} \left[ \nabla f(X_k; \xi) \mid \mathcal{F}_{k-1}^{M,F} \right] &= \nabla f(X_k), \\
\text{Var} \left[ \nabla f(X_k; \xi) \mid \mathcal{F}_{k-1}^{M,F} \right] &\leq V_g.
\end{align*}
\]  

(4.1)

We now discuss how to obtain \( \alpha \)-probabilistically \( \kappa \)-fully linear models and \( \beta \)-probabilistically \( \epsilon_F \)-accurate estimates. In the later analysis, we employ the standard sample averaging approximation technique \cite{33} to construct the sufficient accurate model and estimates. Let

\[
F_k = \frac{1}{|O_k|} \sum_{i \in O_k} f(X_k; \xi_i), \quad G_k = \frac{1}{|O_k|} \sum_{i \in O_k} \nabla f(X_k; \xi_i),
\]

(4.2)

where \( \xi_i \) are the i.i.d. and finite realizations of the noise \( \xi \) and the sample set \( O_k \subseteq \{1, 2, \cdots, n\} \) of size \( |O_k| = p_k \). The local approximation model can be constructed as \( M_k(x) = F_k + G_k^T(x - X_k) + \frac{1}{2}(x - X_k)^T B_k(x - X_k) \), where \( x \in B(X_k, \Delta_k) \). We assume that \( B_k \) is bounded (Assumption 3.2) and it is easy to be satisfied. So here we do not explain how to set \( B_k \). In the following numerical experiments, we will describe how to update \( B_k \) in details.

We recall the definitions in Section 2.1. In order to satisfy Assumption 2.1(i), we need the following conditions hold

\[
\begin{align*}
\mathbb{P}(f(x) - M_k(x) \geq \kappa_{sf} \Delta_k^2 \mid \mathcal{F}_{k-1}^{M,F}) &\leq 1 - \alpha', \\
\mathbb{P}(\|\nabla f(x) - \nabla M_k(x)\| \geq \kappa_{eg} \Delta_k \mid \mathcal{F}_{k-1}^{M,F}) &\leq 1 - \alpha'(\alpha = \alpha^2 \in [0, 1])
\end{align*}
\]  

(4.3)

for all \( x \in B(X_k, \Delta_k) \). By Chebyshev’s inequality in Lemma 5, at current point \( X_k \), for any \( v > 0 \), we have

\[
\mathbb{P} \left[ |F_k - f(X_k)| \geq v \mid \mathcal{F}_{k-1}^{M,F} \right] \leq \frac{V_f}{p_k v^2}, \quad \mathbb{P} \left[ \|G_k - \nabla f(X_k)\| \geq v \mid \mathcal{F}_{k-1}^{M,F} \right] \leq \frac{V_g}{p_k v^2}.
\]  

(4.4)

By (4.4), the conditions (4.3) hold at \( x = X_k \), as long as the sample rate \( p_k \) satisfies the following condition that

\[
p_k \geq \max \left\{ \frac{V_f}{(1 - \alpha') \kappa_{sf}^2 \Delta_k^4}, \frac{V_g}{(1 - \alpha') \kappa_{eg}^2 \Delta_k^2} \right\}.
\]  

(4.5)

For \( \forall x \in B(X_k, \Delta_k) \setminus \{X_k\} \), if \( \|G_k - \nabla f(X_k)\| \leq \kappa_{eg} \Delta_k \), we have

\[
\begin{align*}
\|\nabla M_k(x) - \nabla f(x)\| &= |G_k + B_k(x - X_k) - \nabla f(x)| \\
&= |G_k + B_k(x - X_k) - \nabla f(X_k) + \nabla f(X_k) - \nabla f(x)| \\
&\leq \|G_k - \nabla f(X_k)\| + \|B_k\| \|x - X_k\| \\
&\leq (\kappa_{eg} + L + \kappa_{bhm}) \Delta_k.
\end{align*}
\]  

(4.6)
Thus there exists a constant $x = X_k \leq \Delta_k$, $f$ is $L$-smooth, and the second-order matrix $B_k$ satisfies Assumption 3.2. Thus we can conclude that if the sample rate $p_k$ satisfies (4.5), condition (4.3) will hold for any $x \in B(x_k, \Delta_k)$. Consequently, $M_k(x)$ is $\alpha$-probabilistically $\kappa$-fully linear models. In practice, the function value is not explicitly computed at model $m_k$, so we only require that

$$p_k \geq \frac{V_g}{(1 - \alpha') \kappa^2 g \Delta_k^2}.$$  

Similarly, to obtain $\beta$-probabilistically $\epsilon_F$-accurate estimates, let estimates $\{F^0_k, F^d_k\}$ be

$$F^0_k = \frac{1}{q_k} \sum_{i \in O_k} f(X_k; \xi_i), \quad F^d_k = \frac{1}{q_k} \sum_{i \in O'_k} f(X_k + D_k; \xi_i),$$  

where $O'_k \subseteq \{1, \cdots, N\}$ and $|O'_k| = q_k$. In order to satisfy Assumption 2.1(ii) such that

$$\mathbb{P}(|f(X_k) - F^0_k| \geq \epsilon_F \Delta_k^2 \mid \mathcal{F}^M_{k-1/2}) \leq 1 - \beta', \quad \mathbb{P}(|f(X_k + D_k) - F^d_k| \geq \epsilon_F \Delta_k^2 \mid \mathcal{F}^M_{k-1/2}) \leq 1 - \beta' (\beta^2 = \beta \in [0, 1]),$$

we require that

$$q_k \geq \frac{V_f}{(1 - \beta') \epsilon_F^2 \Delta_k^2} \approx \mathcal{O} \left( \frac{1}{\Delta_k^2} \right).$$  

By the Hölder’s inequality for expectation that $\mathbb{E}[ab] \leq (\mathbb{E}[a^2])^{1/2} (\mathbb{E}[b^2])^{1/2}$ for $a, b > 0$, we have

$$\mathbb{E} \left[ \frac{|F^0_k - f(X_k)|^2}{\epsilon_F^2 \Delta_k^2} \mid \mathcal{F}^M_{k-1/2} \right] \leq \mathbb{E} \left[ \left| f(X_k) \right|^2 \mid \mathcal{F}^M_{k-1/2} \right] \leq \frac{V_f}{q_k \epsilon_F^2 \Delta_k^2} \leq \mathcal{O}(1).$$

Thus there exists a constant $\kappa_f > 0$ such that $\mathbb{E} \left[ |F^0_k - f(X_k)| \mid \mathcal{F}^M_{k-1/2} \right]$ is bounded by $\kappa_f \Delta_k^2$. The similar result can be applied to $F^d_k - f(X_k + D_k)$. We can claim that Assumption 2.1(iii) holds. Thus, we have shown how to construct the random model $m_k$ and estimates $\{F^0_k, F^d_k\}$ to satisfy Assumption 2.1.

However, we can see that the trust-region radius $\Delta_k$ is unknown before the sample rate is chosen as in (4.5) and (4.9). It brings a challenge to construct a fully linear model in practice. First of all, obviously, in the case that the total sample set is very large but limited, such sample rate $p_k$ must exist, such as $p_k = n$, to make the model fully linear. But it is really difficult to give a computable condition of $p_k$ in theory, and prove that under such condition the model $M_k$ is fully linear with probability. Thus we settle for the minimal sample rate of increase in $p_k$ as [34]. As previously discussed in (4.5), the sample size $p_k$ satisfies

$$p_k \geq \frac{V_g}{(1 - \alpha') \kappa^2 g \Delta_k^2} \geq \frac{V_g}{(1 - \alpha') \kappa^2 g \mu_{\max}^2 \|G_k\|^2} \approx \mathcal{O} \left( \frac{V_g}{\|G_k\|^2} \right).$$  

(4.11)
In order to guarantee the fully linear property, from the analysis in Section 3, the critical is that there exists a constant $\kappa_1 > 0$ such that the following condition holds

$$\|G_k - \nabla F(X_k)\| \leq \kappa_1 \|G_k\|. \quad (4.12)$$

The above condition yields

$$\|G_k\| \geq \frac{1}{\kappa_1 + 1} \|\nabla F(X_k)\|. \quad (4.13)$$

In [34], they have analysed gradient-based optimization with dynamic sample sizes for the strongly convex problem. To make the condition (4.12) holds at every iteration, they suggest that the sample size should grow geometrically with iteration $k$, i.e.

$$p_k = p_0 \alpha_1^k \quad \text{for some } \alpha_1 > 1, \ p_0 > 0. \quad (4.14)$$

Intuitively, it make sense because if condition (4.12) holds, $\|\nabla F(X_k)\|^2$ is geometrically decreasing in this case, so is $\|G_k\|^2$. Theorem 3.4 shows that our algorithm achieves the same complexity rate as Theorem 4.1 in [34]. By (4.11), we impose the sample rate $p_k$ to be exponentially increased as (4.14) for the strongly convex case. However, for the nonconvex problem, the situation is different. From (4.11) and (4.13), to obtain an $\epsilon$-solution ($\|\nabla F(X_k)\| \leq \epsilon$), the sample rate $p_k$ should increase with $1/\epsilon^2$. From the analysis in Section 3.2, we have the complexity that the iterations budget $T_\epsilon \approx O(1/\epsilon^2)$. Thus, we have that the sample rate $p_k$ should linearly increase with iteration $k$, i.e.

$$p_k = \min \{\alpha_2 k + p_0, p_{\text{max}}\} \quad (4.15)$$

for some $\alpha_2, p_0 \in \mathbb{R}$ and $p_{\text{max}}$ is a predetermined maximum of $p_k$.

As we know, the situation in practice is very complicated. Thus we require that the sample rate grows exponentially with iteration $k$ for regularized logistic regression problem and linearly increases with iteration $k$ for the deep neural networks, respectively. For more details, please refer to the mini-batch size $p_k$ chosen in Sections 4.2 and 4.3.

### 4.2. Experimental results on regularized logistic regression problem

In this subsection, we consider the following smooth (strongly convex) regularized logistic loss problem considered in [23]

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} \log(1 + \exp(-b_i (a_i^T x))) + \frac{\lambda}{2} \|x\|^2, \ x \in \mathbb{R}^d, \quad (4.16)$$

where $\{(a_i, b_i)\}_{i=1}^{n}$ is a training sample set with $a_i \in \mathbb{R}^d$ being the feature vector and $b_i \in \{-1, +1\}$ being the corresponding label. And $\lambda \geq 0$ is the regularization parameter. As in the typical machine learning setting, we assume that $n$ is very large and $n \gg d$. So computing $f(x)$, as well as $\nabla f(x)$ and $\nabla^2 f(x)$ are very expensive. In our work, we randomly (without replacement) choose a subset $I_k \subseteq \{1, 2, \cdots, n\}$ to estimate the quantities in our algorithms. For the algorithms that need to compute Hessian matrix, the same sample is drawn for gradient and Hessian evaluations. In this setting, we re-sample the sample set for $f_k^1$ and $f_k^2$.

We compare our algorithm STRME with STORM which is implemented as algorithm 5 in [23]. In our numerical experiments, we construct two versions of STRME: one which only computes the stochastic gradients and sets $B_k = 0$ is the first-order version (called STRME-1st), the other one which in addition to the stochastic gradients, computes stochastic Hessian
Table 4.1: Datasets for regularized logistic regression.

<table>
<thead>
<tr>
<th>dataset</th>
<th>N</th>
<th>d</th>
<th>λ</th>
</tr>
</thead>
<tbody>
<tr>
<td>a9a</td>
<td>32561</td>
<td>123</td>
<td>10(^{-4})</td>
</tr>
<tr>
<td>ijcnn1</td>
<td>49990</td>
<td>22</td>
<td>10(^{-4})</td>
</tr>
</tbody>
</table>

estimators is the second-order version (called STRME-2st). We use the classic dogleg method in [35] to solve the second order subproblem, and the corresponding algorithms, we call STRME-dogleg and STORM-dogleg. For the details of the implementation, one can refer to Algorithm C.1 in Appendix. Besides, we compare against a special adaptive solver AdaGrad [2], which takes the adaptive step size but does not have to compute function value, and only computes the average stochastic gradients. For regularized logistic regression, it is easy to get its high-accurate solution. Then we use this solution to get optimal function value \(f^*\). So we choose training loss error \((f(x_k) - f^*)\) as one of criteria to estimate the performance of all algorithms.

For problem (4.16), all algorithms were tested with different input parameters. We set \(x_0 = 0\) as the starting point for all algorithms. For the three quantities gradient, Hessian and \((f_k^0, f_k^d)\) evaluations, we adopt the linearly increased sample rule as in [23] for STORM and our STRME algorithm, i.e. \(p_k = \min \{p_{\text{max}}, \max \{\alpha_2 k + p_0, 1/\delta_{k-1}\}\}\), where \(p_0 = d + 1\), \(p_{\text{max}} = n\), \(\alpha_2 = 100\). In order to achieve good numerical performance, we set the sample rate of function evaluations as \(q_k = p_k\), which is slightly different from the theoretical viewpoint. For AdaGrad, the mini-batch size \(b = p_0 = d + 1\). We set the same random seed to generate random sample sequence for all the algorithms.

For regularized logistic regression problem we consider in this subsection is strongly convex. As we discussed in Section 4.1, to make sure that the model is fully linear, the sample size should be exponentially increased. Thus we also tested the mini-batch size \(p_k = \min \{p_{\text{max}}, \max \{p_0\alpha_1, 1/\delta_{k-1}\}\}\), where \(p_0 = d + 1\), \(p_{\text{max}} = n\), \(\alpha_1 \in \{1.1, 1.3, 1.5, 2\}\) for our algorithm STRME and STORM. We compare the two sample rules as well at the end of this part.

In this subsection, we test on two datasets a9a and ijcnn1 from the LIBSVM website.\(^1\) We list the datasets in Table 4.1, in which \(N\) denotes the total sample size, and \(d\) is the dimension of the dataset, and \(\lambda\) is the regularization parameter. We use 0.95 partition of the data as the training set, and the remaining as the testing set, just like in [23].

In Fig. 4.1, we present the results on the a9a dataset. For STORM-1st and STORM-2st, we use the same parameters in [23]: \(\delta_{\text{max}} = 10, \delta_0 = 1, \gamma = 2, \eta_1 = 0.1, \eta_2 = 0.001\). For STRME-1st and STRME-2st, the following parameters are used: \(\mu_{\text{max}} = 1000, \mu_0 = 1, \eta_1 = 0.1, \gamma = 2\). We set step size \(\eta = 1\) for AdaGrad.

In Fig. 4.2, we report the results on the ijcnn1 dataset. The parameters \(\mu_0 = 10, \mu_{\text{max}} = 10^3, \eta_1 = 0.1, \gamma = 2\) are set for STRME-1st and \(\mu_0 = 1\), others are the same for STRME-dogleg. For STORM-1st and STORM-dogleg, we set \(\delta_0 = 1, \delta_{\text{max}} = 10, \eta_1 = 0.1, \gamma = 2\). For AdaGrad, we set step size \(\eta = 1\).

From Figs. 4.1 and 4.2, we can find that the proposed STRME is comparable to STORM in this setting, both in terms of the training function value and accuracy. The second order methods including STRME-dogleg and STORM-dogleg achieve similar performance as AdaGrad. However, as pointed by a referee, AdaGrad is not the state-of-the-art adaptive gradient method. We have to admit that our current trust region methods cannot outperform the

\(^1\) https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/
state-of-the-art adaptive gradient methods such as Adam and AMSGrad.

At the end of this part, we compare the exponentially increased sample rule with the linearly increased case in Fig. 4.3. We choose the best exponential ratio $\alpha_1 = 1.1$ from $\{1.1, 1.3, 1.5, 2\}$ for all the experiments. The other parameters are the same as the linearly increased sample. For the sake of distinction, we use STRME-L and STRME-E to denote the linearly increased and exponentially increased sample rules respectively, so does STORM. As a whole, from Fig. 4.3, we can see that the linearly increased sample rule performs better than the exponentially increased case. Thus we choose the former one as the sample rule of our experiments.

4.3. Experimental results on a simple deep neural network (DNN)

In this subsection, we consider to train a fully-connected 2-layer net with 50 hidden units (784-50-10) neural networks with MNIST\(^1\) $(n = 60000, d = 784)$, a benchmark dataset of handwritten digits. We used softmax output, sigmoid hidden functions, and the cross-entropy error function. The $l_2$ regularization parameter $\lambda = 10^{-3}$, suggested in [10].

We compare our algorithm STRME with STORM [23]. As in the previous subsection, we construct two versions of STRME: one which only computes the stochastic gradients and set $B_k = 0$ is the first-order version, the other one which in addition to the stochastic gradient, computes the quasi-Newton matrix $B_k$ to approximate the true Hessian matrix is the second-order version. Besides, we make an implementation of the SdLBFGS [36], which is an efficient second-order algorithm for the nonconvex problem, to compare with STRME and STORM.

In our work, we run one epoch mini-batch SGD algorithm to obtain an initial point for all algorithms. In our implementation, we randomly (without replacement) choose the subset $I_k \subseteq \{1, 2, \cdots, n\}$ to estimate the gradient and Hessian pair $(g_k, B_k)$ and objective function value pair $(f_0^k, f_d^k)$. We have attempted the three cases: (i) sample gradient and Hessian pair, and re-sample $f_0^k$ and $f_d^k$ independently; (ii) sample the two pair independently; (iii) sample the two pairs with the same subset. However, for the first two cases, the results are not satisfactory. Therefore, in our numerical experiments, we only consider the last one. Besides, we set the sample rates of gradient and function evaluations as $p_k = q_k$ for the numerical performance.

\[^1\] http://deeplearning.net/data/mnist/

![Results on a9a](image1.png)

![Results on a9a](image2.png)

Fig. 4.1. The results of training regularized logistic regression on a9a.
For STORM and STRME, the mini-batch size $p_k = \min \{ p_{\text{max}}, \max \{ \alpha_2 k + p_0, 1/\delta_k^2 \} \}$, where $p_0 = d + 1$, $p_{\text{max}} = n$, $\alpha_2 = 10$.

4.3.1. Experimental results on the first-order probabilistic model

In this part, we first construct a first-order probabilistic model $m_k$ at $x_k$, i.e.

$$m_k(x_k + d) = f_k + g^T_k d,$$

(4.17)

where $f_k$ and $g_k$ are computed as (4.2), to test the STRME (STRME-1st) framework. Actually, in practice, we do not need to compute $f_k$. In our numerical experiments, we compare the numerical performance of STRME-1st with the related STORM-1st.

We now give details of parameters in the proposed STRME-1st and STORM-1st. For STRME-1st, $\mu_0$ and $\mu_{\text{max}}$ are two important parameters. The parameters $\mu_0$ is chosen from \{0.01, 0.1, 1\}, and the best $\mu_0$ is achieved at $\mu_0 = 0.1$. Compared to $\mu_0$, the parameter $\mu_{\text{max}}$ is more important in the later iteration process. Therefore the range of $\mu_{\text{max}}$ is more elaborate. In
our implementation, let $\mu_{\text{max}} \in \{0.5, 1, 2, 4, 8\}$, and the best tuned $\mu_{\text{max}}$ is achieved at $\mu_{\text{max}} = 2$. For STORM-1st, we test on $\delta_0 \in \{0.01, 0.1, 1\}$ and $\delta_{\text{max}} \in \{0.5, 1, 2, 4, 8\}$, the best performance is achieved with the inputs $(\delta_0, \delta_{\text{max}}) = (0.1, 1)$. The results are showing in Fig. 4.4.

From Fig. 4.4, we can see that STRME-1st is better than STORM-1st with the model constructed as the beginning of this section. Not surprisingly, the trust-region radius constructed by STRME can make better use of gradient information. However, we can construct a more efficient second-order model to make the STRME algorithm perform better.

4.3.2. Experiments on second-order probabilistic models

In this part, we construct a specific second-order probabilistic model and implement STRME algorithm framework in deep neural network. In our work, we consider the limited memory symmetric rank one method (L-SR1) to generate the second order quasi-Newton matrix $B_k$,
and build up the second-order model \( m_k(x_k + d) \) as follows
\[
m_k(x_k + d) = f_k + g_k^T d_k + \frac{1}{2} d^T B_k d_k,
\]
where \( f_k \) and \( g_k \) are defined as the beginning of this section. Next, we will show how to update the quasi-Newton matrix \( B_k \).

Let \( s_k \triangleq x_{k+1} - x_k \) and \( y_k \triangleq \frac{1}{m_k} \left( \sum_{j=1}^{m_k} \nabla f_{i_j}(x_{k+1}) - \nabla f_{i_j}(x_k) \right) \). Given an initial matrix \( B_0 \), provided that \((y_k - B_k s_k)^T s_k \neq 0\), then \( B_{k+1} \) can be defined as
\[
B_{k+1} \triangleq B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}.
\]

Limited memory symmetric rank one (SR1) method stores and uses the \( m \) most recently computed pair \( \{(s_k, y_k)\} \) \((m \ll d)\). To describe the compact representation of a L-SR1 matrix, we need to define, for \( k \geq m \)
\[
S_k \triangleq [s_{k-m+1}, s_{k-m}, \ldots, s_k] \in \mathbb{R}^{d \times m},
\]
\[
Y_k \triangleq [y_{k-m+1}, y_{k-m}, \ldots, y_k] \in \mathbb{R}^{d \times m}.
\]

Moreover, we need the following decomposition of \( S_k^T Y_k \)
\[
S_k^T Y_k = L_k + D_k + R_k,
\]
where \( L_k \) is strictly lower triangular, \( D_k \) is diagonal, and \( R_k \) is strictly upper triangular. We assume that all the updates are well-defined, i.e. \( s_k^T (y_k - B_k s_k) \neq 0 \), otherwise we skip the update. The compact form of L-SR1 [37] can be written as
\[
B_{k+1} = B_0 + U_k V_k U_k^T,
\]
where \( U_k \in \mathbb{R}^{d \times m} \), \( V_k \in \mathbb{R}^{m \times m} \), and \( B_0 \) is a diagonal matrix. \( U_k \) and \( V_k \) are given by
\[
U_k = Y_k - B_0 S_k \quad \text{and} \quad V_k = (D_k + L_k + L_k^T - S_k^T B_0 S_k)^{-1}.
\]
Now the quadratic probabilistic model defined by the L-SR1 method is constructed. The trust-region subproblem will be
\[
\min_{\|d\| \leq \delta_k} m_k(x_k + d) = f_k + g_k^T d + \frac{1}{2} d^T B_k d.
\] (4.24)

Applying this model into STRME, we can obtain STRME-LSR1 algorithm. Similarly, we can obtain STORM-LSR1 algorithm. With respect to the specific implementation of the subproblem and how to solve the trust-region subproblem efficiently, one can refer to the OBS method in [38].

In this part, we compare our algorithm STRME-LSR1 with STORM-LSR1 and its first-order form. The result is shown in Fig.4.5. We test on different choice of matrix \( B_0 \): (i) \( B_0 = s_0^T y_0 / (s_0^T s_0) \|_d \); (ii) \( B_0 = y_0^T y_0 / (s_0^T y_0) \|_d \); (iii) \( B_0 = \tau_0 \|_d \), where \( \tau_0 \) is adjusted with the same scale as \( y_0^T y_0 / (s_0^T y_0) \). Unfortunately, the first two choices are not satisfactory. Thus, we use (iii) to define the initial matrix \( B_0 \).
In our numerical experiment, $B_0 = \tau_0 I_d$, where $\tau_0 \in \{0.5, 1, 1.5\}$, and the best $\tau_0$ is achieved at $\tau_0 = 1$. We have tested the limited memory size $m \in \{10, 20, 30, 40\}$. We find that $m = 30$ is performed relatively better. The parameters $\mu_0$ is chosen from $\{0.01, 0.1, 1\}$, and the best $\mu_0$ is achieved at $\mu_0 = 0.1$. For $\mu_{\text{max}}$, we test on the range $\{0.1, 1, 10, 100\}$, the best one is $\mu_{\text{max}} = 10$, which implies that unlike the first-order STRME, the second-order STRME is not so sensitive to $\mu_{\text{max}}$ and allow some larger values. For STORM-LSR1, we test the $\delta_0$ and $\delta_{\text{max}}$ with the same range as $\mu_0$ and $\mu_{\text{max}}$, the best choices are achieved at $\delta_0 = 0.1, \delta_{\text{max}} = 1$, respectively. From Fig. 4.5, we can see that our algorithm STRME-LSR1 performs better than STORM-LSR1. And it is not hard to find that STRME-1st is not worse than STORM-LSR1.

Moreover, in Fig. 4.6, we show the behavior of successful and failed ratio $\varsigma$ which denotes the total number of successful iterations divided by the total number of failed iterations. In this case, we set the maximum number of gradient $SFO_{\text{max}} = 100n$. For the first-order methods, we can see that the ratio $\varsigma$ of STORM-1st is basically stable around 1, and the ratio of STRME-1st is higher and increasing in the later period. Besides, we can see that in the middle and later period of the algorithm STRME-LSR1 and STORM-LSR1, the value of $\varsigma$ is basically stable. However, for STRME-LSR1, the value of $\varsigma$ in stable condition is still larger than that for STORM-LSR1.

Beyond that we also compare the trust-region radius $\delta_k$ of the two algorithms with a long time training in Fig. 4.7, where $SFO_{\text{max}} = 100n$. We can see that for STRME, whether STRME-1st or STRME-LSR1, the bandwidth of trust-region radius is narrower than that of STORM. This means the oscillation of the trust-region radius for STRME is less severe in contrast to that of STORM methods. Besides, for STRME-LSR1, $\delta_k$ is overall declining and smaller than that in STORM-LSR1. At the same time, we observe that the second-order methods permit larger trust-region radius from the Fig. 4.7.

It is worth to mention that the sample size $p_k$ is considerably smaller than $n$ for a long time training. For example: for the first-order trust region methods including STORM-1st and STRME-1st, the rate $p_k/n$ is around 0.14 and 0.07 (STRME-1st), respectively. For the second order trust region methods (STORM-LSR1 and STRME-LSR1), the rate $p_k/n$ is close 0.36 and 0.18 (STRME-LSR1) respectively at the final stage of the iteration.

Moreover, we implement the well-known second-order algorithm SdLBFGS [36] to test the performance of our algorithm in long time running ($SFO_{\text{max}} = 100n$). In this case, we set batch size $b = d + 1$ and step size $\eta_k = \beta_0/(k/10 + 1)$ where $\beta_0 \in \{0.1, 1, 10\}$ for SdLBFGS. The best tuned step size is obtained at $\beta_0 = 10$. In [36], they set $\eta_k = 10/(k + 1)$. By numerical comparison, we find that the result of $\eta_k = 10/(k/10 + 1)$ is better than that for the choice $\eta_k = 10/(k + 1)$. The results in Fig. 4.8 illustrate that STRME-LSR1 performs better than the best tuned SdLBFGS.

### 5. Conclusion

We have presented a stochastic trust-region framework in which the trust-region radius depends on the currently probabilistic model. To verify the effectiveness of the framework, we have proposed a specific algorithm named STRME in which the trust-region radius is linearly associated with the model gradient. We have analysed the expected number of iterations of STRME for three different cases: nonconvex, convex, and strongly convex. We can see that our algorithm enjoys the same complexity properties as the existing schemes. Moreover, our algorithm compares favorably to STORM algorithm and other stochastic algorithms on several
testing problems involving the real datasets. Actually, in addition to STRME, there are many other approaches to explore the trust-region radius related with random models. We point out that the work in this paper is limited to the case that the objective function is smooth. There are some important and latest works for non-smooth problems, for example [39–41] and [42]. It is worthwhile to extend the stochastic trust-region framework to the non-smooth setting.

Moreover, the effectiveness of the stochastic trust-region method is relevant to the model. In our experiments, we used a simple sample averaging technique to reduce the variance of the gradient or function values when we construct the probabilistic models. If the variance reduction technique like SVRG [6] is applied to reduce the variance, it may be greatly reduce

\[ p_k \text{ and } q_k \]

It will be an interesting subject for future research.

**Appendix A. Analysis of the Stochastic Process \{\Lambda_k, \Phi_k\}**

In this part, we aim to estimate the upper bound of an expected stopping time by observing the behavior of stochastic process generated by Algorithm 2.1. The results can be applied to analyse the convergence rates of the proposed algorithm in different settings. We first give some basic definitions before theoretical analysis.

**Definition A.1.** Let \( X = \{X_k, k \geq 0\} \) be a stochastic process. We say \( T \) is a stopping time with respect to \( X \) if for each \( k > 0 \), the event \( \{T = k\} \) is completely determined by the total information up to time \( k \), i.e., \( \{X_0, X_1, \cdots, X_k\} \).

Here we give a random variable \( T_\epsilon \), which is the total number of iterations until an \( \epsilon \)-solution is achieved.

**Remark A.1.** \( T_\epsilon \) is a special stopping time and dependent on randomness of the proposed algorithm and the \( \epsilon \)-solution we have defined.

Next, we consider a stochastic process \( \{\Lambda_k, \Phi_k\} \) such that \( \Lambda_k \in [0, \infty) \) and \( \Phi_k \in [0, \infty) \) for all \( k > 0 \). Let us give the definition of a special random event \( W_k \) as follows

\[
\mathbb{P}(W_k = 1|\mathcal{F}_{k-1}^M) = p, \quad \mathbb{P}(W_k = -1|\mathcal{F}_{k-1}^M) = 1 - p, \quad (A.1)
\]

where \( p \in [0, 1] \). We now assume that \( \Lambda_k \) and \( \Phi_k \) satisfy the assumption mentioned below, for all \( k < T_\epsilon \).

**Assumption A.1.**

1. There exist constants \( \lambda \in (0, \infty) \) and \( \mu_{\max} = \Lambda_0 e^{\lambda j_{\max}} > 0 \) (for some \( j_{\max} \in \mathbb{Z} \)) such that \( \Lambda_k \leq \mu_{\max} \).

2. There exists a constant \( \hat{\Lambda} = \Lambda_0 e^{\lambda j_{\epsilon}} > 0 \) (for some \( j_{\epsilon} \in \mathbb{Z} \)) such that for all \( k \leq T_\epsilon \), the following properties hold

\[
\Lambda_{k+1} \geq \min(\Lambda_k e^{\lambda_1 W_k}, \hat{\Lambda}),
\]

where \( \lambda_1 \in \mathbb{R} \) and \( W_k \) satisfies (A.1) with \( p > \frac{1}{2} \).

3. There exists a constant \( C > 0 \), and a non-decreasing function \( h(\cdot) \) which is positive on any positive domain, such that for all \( k < T_\epsilon \),

\[
\mathbb{E}\left[ \Phi_{k+1}|\mathcal{F}_{k-1}^M \right] \leq \Phi_k - Ch(\Lambda_k).
\]
This assumption is not related with the convexity of the objective function, so it holds in all three cases, i.e. nonconvex case, convex case and strongly convex case. Based on Assumptions A.1, the following theorem (see [24]) illustrates the upper bound on the expected number of iterations $T_\epsilon$ for obtaining an $\epsilon$-solution.

**Theorem A.1.** Let Assumption A.1 holds. Then

$$E[T_\epsilon] \leq \frac{p}{2p-1} \frac{\Phi_0}{Ch(\Lambda)} + 1.$$  

The analysis of the renewal-reward martingale process in [24] is appropriate for the stochastic process $\{\Lambda_k, \Phi_k\}$ generated by Algorithm 2.1. A slightly difference is that we use $\Lambda_k$ instead of $\Delta_k$ [24] to analyse the random process. So here we omit the proof of Theorem A.1. For more details, we refer the readers to Theorem 2 in [24]. Theorem A.1 is very important to the analysis of the complexity of Algorithm 3.1. The difficulties lie in finding the non-decreasing function $h(\cdot)$ and the constant $\Lambda$.

**Appendix B. Proofs of Some Basic Lemmas and Theorems in Section 3**

**Proof.** (of Lemma 3.1) From Assumption 3.1, the trial step $d_k$ will lead a sufficient reduction on $m_k$ such that

$$m_k(x_k) - m_k(x_k + d_k) \geq \frac{\kappa_{fed}}{2} \|g_k\| \min \left\{ \frac{\|g_k\|}{\|B_k\|}, \delta_k \right\}. \tag{B.1}$$

Since $\delta_k = \mu_k \|g_k\|$ and Assumption 3.2 holds that $\|B_k\| \leq \kappa_{bhm}$, which together with (B.1), implies that

$$m_k(x_k) - m_k(x_k + d_k) \geq \frac{\kappa_{fed}}{2} \|g_k\|^2 \min \left\{ \frac{1}{\kappa_{bhm}}, \mu_k \right\}. \tag{B.2}$$

If the model $m_k$ is true, we can obtain that

$$f(x_k) - f(x_k + d_k)$$

$$= f(x_k) - m_k(x_k) + m_k(x_k) - m_k(x_k + d_k) + m_k(x_k + d_k) - f(x_k + d_k)$$

$$\geq -2\kappa_{ef} \|g_k\|^2 \mu_k^2 + \frac{\kappa_{fed}}{2} \|g_k\|^2 \min \left\{ \frac{1}{\kappa_{bhm}}, \mu_k \right\}$$

$$= \left( -2\kappa_{ef} \mu_k^2 + \frac{\kappa_{fed}}{2} \min \left\{ \frac{1}{\kappa_{bhm}}, \mu_k \right\} \right) \|g_k\|^2.$$ 

Next, we estimate the upper bound of $\mu_k$ to satisfy that

$$-2\kappa_{ef} \mu_k^2 + \frac{\kappa_{fed}}{2} \min \left\{ \frac{1}{\kappa_{bhm}}, \mu_k \right\} \geq \frac{\kappa_{fed}}{4} \min \left\{ \frac{1}{\kappa_{bhm}}, \mu_k \right\}. \tag{B.3}$$

If $\mu_k \leq 1/\kappa_{bhm}$, in order to satisfy (B.3), we have

$$\mu_k \leq \frac{\kappa_{fed}}{8\kappa_{ef}}.$$
Assume that model \( m \) is true, which means, for all \( y \in B(x_k, \delta_k) \), we have

\[
\| \nabla f(y) - \nabla m_k(y) \| \leq \kappa_{eg} \delta_k,
\]

and

\[
|f(y) - m_k(y)\| \leq \kappa_{ef} \delta_k^2.
\]

Thus, if \( \kappa_{ef} / (8\kappa_{ef} \kappa_{hhm}) \geq 1/\kappa_{hhm} \), we have \( \mu_k \leq \sqrt{\kappa_{ef} / (8\kappa_{ef} \kappa_{hhm})} \); else we have \( \mu_k \leq \kappa_{ef} / (8\kappa_{ef}) \).

We can conclude that no matter whether \( \mu_k \leq 1/\kappa_{hhm} \) or not, as long as \( \mu_k \) satisfy that

\[
\mu_k \leq \min \left\{ \frac{\kappa_{ef}}{8\kappa_{ef}}, \sqrt{\frac{\kappa_{ef}}{8\kappa_{ef} \kappa_{hhm}}} \right\},
\]

we have

\[
f(x_k) - f(x_k + d_k) \geq \frac{\kappa_{ef}}{4} \min \left\{ \frac{1}{\kappa_{hhm}}, \mu_k \right\} \| g_k \|^2.
\]

Thus, the desired result is proved. \( \square \)

\textbf{Proof.} (of Lemma 3.2) If \( d_k \) is accepted, which implies that

\[
f_k^0 - f_k^d \geq \eta_1(m_k(x_k) - m_k(x_k + d_k))
\]

\[
\geq \eta_1 \frac{\kappa_{ef}}{2} \| g_k \| \min \left\{ \frac{\| g_k \|}{\| B_k \|}, \delta_k \right\}
\]

\[
\geq \eta_1 \frac{\kappa_{ef}}{2} \| g_k \|^2 \min \left\{ \frac{1}{\kappa_{hhm}}, \mu_k \right\}
\]

where the last inequality follows from \( \delta_k = \mu_k \| g_k \| \) and \( \| B_k \| \leq 1/\kappa_{hhm} \).

If the estimates \( \{f_k^0, f_k^d\} \) are tight, the improvement in \( f \) can be bounded by

\[
f(x_k) - f(x_k + d_k) = f(x_k) - f_k^0 + f_k^0 - f_k^d + f_k^d - f(x_k + d_k)
\]

\[
\geq -2 \epsilon_F \delta_k^2 + \eta_1 \frac{\kappa_{ef}}{2} \| g_k \|^2 \min \left\{ \frac{1}{\kappa_{hhm}}, \mu_k \right\}
\]

\[
\geq \left( -2 \epsilon_F \mu_k^2 + \frac{\eta_1 \kappa_{ef}}{2} \min \left\{ \frac{1}{\kappa_{hhm}}, \mu_k \right\} \right) \| g_k \|^2
\]

\[
\geq \frac{\eta_1 \kappa_{ef}}{4} \min \left\{ \frac{1}{\kappa_{hhm}}, \mu_k \right\} \| g_k \|^2.
\]

Since \( \epsilon_F \leq \frac{\eta_1 \kappa_{ef}}{8\mu_{max}} \min \left\{ \frac{1}{\kappa_{hhm} \mu_{max}}, 1 \right\} \), we have \( \epsilon_F \mu_k^2 \leq \frac{\eta_1 \kappa_{ef}}{8} \min \left\{ \frac{1}{\kappa_{hhm}}, \mu_k \right\} \) which deduces the last inequality. \( \square \)

\textbf{Proof.} (of Lemma 3.3) Because \( \| B_k \| \leq \kappa_{hhm} \), the Cauchy decrease condition yields

\[
m_k(x_k) - m_k(x_k + d_k)
\]

\[
\geq \frac{\kappa_{ef}}{2} \| g_k \| \min \left\{ \frac{\| g_k \|}{\| B_k \|}, \delta_k \right\} \geq \frac{\kappa_{ef}}{2} \min \left\{ \frac{1}{\kappa_{hhm}}, \mu_k \right\} \| g_k \|^2.
\]

Assume that model \( m_k \) are true, which means, for all \( y \in B(x_k, \delta_k) \), we have

\[
\| \nabla f(y) - \nabla m_k(y) \| \leq \kappa_{eg} \delta_k,
\]

and

\[
|f(y) - m_k(y)\| \leq \kappa_{ef} \delta_k^2.
\]
The estimates \( f_k^0 \) and \( f_k^d \) are tight with \( \epsilon_F \leq \kappa_e f \), we have
\[
|f_k^0 - f(x_k)| \leq \kappa_e f \delta_k^2, \quad |f_k^d - f(x_k + d_k)| \leq \kappa_e f \delta_k^2.
\]
(B.6)
The \( f_k^0 - f_k^d \) can be rewritten as
\[
f_k^0 - f_k^d = f_k^0 - f(x_k) + f(x_k) - m_k(x_k) + m_k(x_k) - m_k(x_k + d_k) + m_k(x_k + d_k) - f(x_k + d_k) + f(x_k + d_k) - f_k^d.
\]
(B.7)
Applying the inequalities (B.5) and (B.6) to the above equality, we can obtain
\[
|f_k^0 - f_k^d - (m_k(x_k) - m_k(x_k + d_k))| \leq 4 \kappa_e f \delta_k^2
\]
(B.8)
Since \( \mu_k \leq \min \left\{ \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}, \sqrt{\frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}\right\} \), and applying (B.4) and using \( \delta_k = \mu_k \|g_k\| \), then we have
\[
4 \kappa_e f \delta_k^2 \leq (1 - \eta_1)(m_k(x_k) - m_k(x_k + d_k)).
\]
(B.9)
Thus, from (B.8) and (B.9), we can conclude that \( f_k^0 - f_k^d \geq \eta_1 (m_k(x_k) - m_k(x_k + d_k)) \), which means that the \( k \)-th iteration is successful.

To estimate the complexity of Algorithm 3.1, we aim to bound the expected number of iterations \( E[T_e] \). Before that, we have to prove that Assumption A.1 holds for the process \( \{\Lambda_k, \Phi_k\} \) generated by Algorithm 3.1, where \( \Phi_k \) is defined as (3.3). Let us define the constant \( \Lambda \) in Assumption A.1(2) as follows:
\[
\Lambda = \zeta, \quad \text{where} \quad \zeta \leq \min \left\{ \mu_{\max}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}, \sqrt{\frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}\right\}.
\]
(B.10)
For simplicity, we assume that \( \Lambda_0 = \gamma^i \Lambda \) and \( \mu_{\max} = \gamma^j \Lambda \) for some integers \( i, j > 0 \). As a result, for any \( k > 0 \), we have \( \Lambda_k = \gamma^k \Lambda \) for some integer \( i \). It is apparent that Assumption A.1(1) holds with \( \mu_k \leq \mu_{\max} \) in Algorithm 3.1. Next, we will show that Assumption A.1(2) holds provided the constant \( \Lambda \) is defined as above.

Lemma B.1. Let \( \alpha \) and \( \beta \) satisfy that \( \alpha \beta > \frac{1}{2} \), then we have Assumption A.1(2) holds with \( W_k = 2(I_k J_k - \frac{1}{2}) \), \( \lambda_1 = \log(\gamma) \), and \( p = \alpha \beta \).

Proof. First, we show that for all \( k < T_e \), the following inequality holds
\[
\Lambda_{k+1} \geq \min \left\{ \Lambda, \min \{\mu_{\max}, \gamma \Lambda_k\} I_k J_k + \frac{1}{\gamma} \Lambda_k (1 - I_k J_k) \right\}.
\]
(B.11)
If \( \Lambda_k \geq \Lambda \), we have \( \Lambda_{k+1} \geq \gamma \Lambda \) by the update process of the sequence \( \Lambda_k \). Hence, \( \Lambda_{k+1} \geq \Lambda \).

Now we assume that \( \Lambda_k \leq \Lambda \), by the definition of \( \Lambda \), we have
\[
\Lambda_k \leq \min \left\{ \mu_{\max}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}, \sqrt{\frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}, \frac{\kappa_{fed}(1-\eta_1)}{8 \kappa_e f}\right\}.
\]
(B.12)
If \( I_k = 1 \) and \( J_k = 1 \), i.e. model \( M_k \) and estimates are all sufficiently accurate, from Lemma 3.3, we know that the iteration \( k \) is successful. Thus \( X_{k+1} = X_k + D_k \) and \( \Lambda_{k+1} = \gamma \Lambda_k \). If \( I_k J_k = 0 \), whether the iteration \( k \) is successful or failed, we all have \( \Lambda_{k+1} \geq \frac{1}{\gamma} \Lambda_k \).
From the above analysis, we conclude that (B.11) holds. Because

\[
E \left[ I_k = 1 | F_{k-1}^{M,F} \right] \geq \alpha,
E \left[ J_k = 1 | F_{k-1/2}^{M,F} \right] \geq \beta,
\]
we have

\[
E \left[ I_k J_k = 1 | F_{k-1}^{M,F} \right] = E \left[ J_k = 1 | I_k = 1, F_{k-1}^{M,F} \right] \cdot E \left[ I_k = 1 | F_{k-1}^{M,F} \right] \geq \alpha \beta.
\] (B.13)
So \( p = P(W_k = 1 | F_{k-1}^{M,F}) = P(I_k J_k = 1 | F_{k-1}^{M,F}) \geq \alpha \beta > \frac{1}{2} \), which implies that Assumption A.1(2) holds. \( \square \)

Using the above Lemmas 3.2 and 3.3, we can derive the result of Theorem 3.1.

**Proof.** (of Theorem 3.1) Firstly, we recall the definition of \( \Phi_k \) as (3.3), i.e.

\[
\Phi_k = \nu(f(X_k) - f^*) + (1 - \nu) \frac{1}{2} \Lambda_k \| \nabla f(X_k) \|^2.
\]
In the following proof, we consider three separate cases: (i) model \( M_k \) is true and estimates \( \{ F_k^0, F_k^d \} \) are tight \( (I_k = J_k = 1) \); (ii) model \( M_k \) is false and estimates \( \{ F_k^0, F_k^d \} \) are tight \( (J_k = 1, I_k = 0) \); (iii) estimates \( \{ F_k^0, F_k^d \} \) are loose \( (J_k = 0) \). For each of these cases, we will analyse two possible outcomes of the iteration process, i.e. the iteration \( k \) is successful or failed.

Based on the above classifications, we rewrite the expected decrease of \( \Phi_k \) as

\[
E \left[ \Phi_{k+1} - \Phi_k | F_{k-1}^{M,F} \right] = E \left[ (I_{(i_k J_k=1)} + I_{((1-i_k)J_k=1)} + I_{((1-J_k)=1)})(\Phi_{k+1} - \Phi_k) | F_{k-1}^{M,F} \right].
\]

Before presenting the formal proof, we briefly describe the key ideas. By choosing a suitable constant \( \nu \), we can derive an upper bound on the expected decrease of \( \Phi_k \) for each of the cases. When model \( M_k \) is true and estimates \( \{ F_k^0, F_k^d \} \) are tight, no matter whether the iteration \( k \) is successful or not, it will give rise to the decrease of \( \Phi_k \) which is in proportion to \( \Lambda_k \| \nabla f(X_k) \|^2 \).

For the other two cases, due to the model error or inaccurate estimates, \( \Phi_k \) may increase. However, the increment of \( \Phi_k \) is still in proportion to \( \Lambda_k \| \nabla f(X_k) \|^2 \). Therefore, by choosing sufficiently large \( \alpha \in (0, 1) \) and \( \beta \in (0, 1) \), the expectation of \( \Phi_k \) can be guaranteed to decrease.

(ii) **Model \( M_k \) is true and estimates \( \{ F_k^0, F_k^d \} \) are tight \( (I_k = J_k = 1) \).**

(a) **Iteration \( k \) is successful.** In this case, we have \( X_{k+1} = X_k + D_k \), and \( \Lambda_{k+1} = \gamma \Lambda_k \).

Because the estimate \( \{ F_k^0, F_k^d \} \) are tight. From Lemma 3.2, if

\[
\epsilon_F \leq \frac{\eta_f K_{fcd}}{\delta \mu_{\max}} \min \left\{ \frac{1}{\kappa_{bh,m} \mu_{\max}}, 1 \right\},
\]
we have

\[
f(X_k + D_k) - f(X_k) \leq -\frac{\eta_f K_{fcd}}{4} \min \left\{ \frac{1}{\kappa_{bh,m}}, \Lambda_k \right\} \| G_k \|^2.
\] (B.14)
Besides, since the model \( M_k \) is true, we can easily find the relation between \( \| G_k \| \) and \( \| \nabla f(X_k) \| \), i.e.

\[
\| \nabla f(X_k) - G_k \| \leq \kappa_{eg} \delta_k = \kappa_{eg} \Lambda_k \| G_k \|.
\]
Using the above inequality, the triangle inequality and the fact that \( \Lambda_k \leq \mu_{\max} \), we obtain

\[
\| G_k \| \geq \frac{1}{1 + \kappa_{eg} \mu_{\max}} \| \nabla f(X_k) \|.
\] (B.15)
If $f$ is $L$-smooth and the iteration $k$ is successful, we have
\[ \|\nabla f(X_{k+1}) - \nabla f(X_k)\| \leq L\Delta_k. \] (B.16)

Using the fact that $(a + b)^2 \leq 2(a^2 + b^2)$ and $\Delta_k = \Lambda_k \|G_k\|$, by a simple calculation, (B.16) proceeds that
\[ \|\nabla f(X_{k+1})\|^2 \leq 2 \left( L^2 \Lambda_k^2 \|G_k\|^2 + \|\nabla f(X_k)\|^2 \right). \] (B.17)

Combining (B.17), the following holds with $\Lambda_{k+1} = \gamma \Lambda_k$ and $\Lambda_k \leq \mu_{\text{max}}$
\[ \frac{1}{L^2} \left( \Lambda_{k+1} \|\nabla f(X_{k+1})\|^2 - \Lambda_k \|\nabla f(X_k)\|^2 \right) \leq 2\gamma \Lambda_k \left( \mu_{\text{max}}^2 \|G_k\|^2 + \frac{1}{L^2} \|\nabla f(X_k)\|^2 \right). \] (B.18)

We note that (B.18) holds when $f$ is $L$-smooth and the iteration $k$ is successful, no matter whether the model $M_k$ is true or not. We will reuse (B.18) in the following analysis of cases (ii) and (iii).

Applying (B.14) and (B.18), the increment of $\Phi_k$ is bounded by
\[ \Phi_{k+1} - \Phi_k \leq \nu (f(X_{k+1} - f(X_k))) + (1 - \nu) \frac{1}{L^2} \left( \Lambda_{k+1} \|\nabla f(X_{k+1})\|^2 - \Lambda_k \|\nabla f(X_k)\|^2 \right) \leq -\frac{\nu \eta f_{\text{cd}}}{4} \min \left\{ \frac{1}{\kappa_{\text{bhm}}}, \Lambda_k \right\} \|G_k\|^2 + 2(1 - \nu)\gamma \Lambda_k \left( \mu_{\text{max}}^2 \|G_k\|^2 + \frac{1}{L^2} \|\nabla f(X_k)\|^2 \right). \] (B.19)

We choose $\nu \in (0, 1)$ to satisfy
\[ \frac{1 - \nu}{\nu} \leq \frac{\eta f_{\text{cd}}}{16\gamma \mu_{\text{max}}} \min \left\{ \frac{1}{\kappa_{\text{bhm}} \mu_{\text{max}}}, 1 \right\}, \]
which implies that
\[ -\frac{\nu \eta f_{\text{cd}}}{4} \min \left\{ \frac{1}{\kappa_{\text{bhm}}}, \Lambda_k \right\} + 2(1 - \nu)\gamma \mu_{\text{max}}^2 \Lambda_k \leq -\frac{\nu \eta f_{\text{cd}}}{8} \min \left\{ \frac{1}{\kappa_{\text{bhm}}}, \Lambda_k \right\}. \]

Consequently, (B.19) can be written as
\[ \Phi_{k+1} - \Phi_k \leq -\frac{\nu \eta f_{\text{cd}}}{8} \min \left\{ \frac{1}{\kappa_{\text{bhm}}}, \Lambda_k \right\} \|G_k\|^2 + 2(1 - \nu)\gamma \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2. \] (B.20)

Incorporating the relation of $\|G_k\|$ and $\|\nabla f(X_k)\|$, shown as (B.15), into (B.20), we have
\[ \Phi_{k+1} - \Phi_k \leq \left[ -\frac{\nu \eta f_{\text{cd}}}{8(1 + \kappa_{\text{eg}} \mu_{\text{max}})^2} \right] \|\nabla f(X_k)\|^2. \] (B.21)
Furthermore, we assume that $\nu$ satisfies
\[
\frac{1 - \nu}{\nu} \leq \frac{\eta_1 K_{\text{fcd}} L^2 \min \left\{ \frac{1}{\kappa_{\text{bhm}} m_{\text{max}}}, 1 \right\}}{32 \gamma (1 + \kappa_{\text{eg}} m_{\text{max}})^2},
\]
which yields
\[
- \frac{\nu \eta_1 K_{\text{fcd}} \min \left\{ \frac{1}{\kappa_{\text{bhm}} m_{\text{max}}}, \Lambda_k \right\}}{8(1 + \kappa_{\text{eg}} m_{\text{max}})} + 2(1 - \nu)\gamma \frac{1}{L^2} \Lambda_k \leq - \frac{\nu \eta_1 K_{\text{fcd}} \min \left\{ \frac{1}{\kappa_{\text{bhm}} m_{\text{max}}}, \Lambda_k \right\}}{16(1 + \kappa_{\text{eg}} m_{\text{max}})^2}. \tag{B.22}
\]

Combining (B.21) and (B.22), we have that
\[
\Phi_{k+1} - \Phi_k \leq - \frac{\nu \eta_1 K_{\text{fcd}} \min \left\{ \frac{1}{\kappa_{\text{bhm}} m_{\text{max}}}, \Lambda_k \right\} \|\nabla f(X_k)\|^2}{16(1 + \kappa_{\text{eg}} m_{\text{max}})^2}. \tag{B.23}
\]

Choosing a suitable $\nu \in (0, 1)$ satisfies that
\[
\frac{1 - \nu}{\nu} \leq \frac{K_{\text{fcd}} \eta_1 L^2 \min \left\{ \frac{1}{\kappa_{\text{bhm}} m_{\text{max}}}, 1 \right\}}{16(1 + \kappa_{\text{eg}} m_{\text{max}})^2(1 - \frac{1}{\gamma})}. \tag{B.24}
\]

such that
\[
- \frac{\nu \eta_1 K_{\text{fcd}} \min \left\{ \frac{1}{\kappa_{\text{bhm}} m_{\text{max}}}, \Lambda_k \right\}}{16(1 + \kappa_{\text{eg}} m_{\text{max}})^2} \leq -(1 - \nu)(1 - \frac{1}{\gamma}) \frac{1}{L^2} \Lambda_k,
\]
which means that a successful iteration can achieve a larger reduction on $\Phi_k$ compared to the case that iteration $k$ is failed. Then, no matter whether the iteration is successful or failed, we have
\[
\mathbb{E}\left[ \Phi_{k+1} - \Phi_k \middle| J_{k+1} = 1 \right] \leq \mathbb{E}\left[ \Phi_{k+1} - \Phi_k \middle| J_{k+1} = 0 \right] \leq -(1 - \nu)(1 - \frac{1}{\gamma}) \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2.
\]

Taking conditional expectation on $F_{k-1}^{M,F}$, we have
\[
\mathbb{E}\left[ \Phi_{k+1} - \Phi_k \middle| F_{k-1}^{M,F} \right] \leq \mathbb{E}\left[ \Phi_{k+1} - \Phi_k \middle| J_{k+1} = 1 \right] \leq \mathbb{E}\left[ \Phi_{k+1} - \Phi_k \middle| J_{k+1} = 0 \right] \leq -(1 - \nu)(1 - \frac{1}{\gamma}) \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2. \tag{B.25}
\]

(ii) Model $M_k$ is false and estimates $\{F^0_k, F^d_k\}$ are tight ($J_k = 1, I_k = 0$).

(a) Iteration $k$ is successful. Because the iteration is successful, we have $X_{k+1} = X_k + D_k$, and $\Lambda_{k+1} = \gamma \Lambda_k$. In this case, the estimates $\{F^0_k, F^d_k\}$ are tight. Applying Lemma 3.2, we have
\[
f(X_{k+1}) - f(X_k) \leq - \frac{\eta_1 K_{\text{fcd}}}{4} \min \left\{ \frac{1}{\kappa_{\text{bhm}}}, \Lambda_k \right\} \|G_k\|^2, \tag{B.26}
\]

with \( \epsilon_F \leq \min \{ \eta_1 \kappa_{fcd} / (8 \kappa_{bhm} \mu_{\max}^2), \eta_1 \kappa_{fcd} / (8 \mu_{\max}) \} \). Due to the facts that \( k \) is successful and \( f \) is \( L \)-smooth, so the inequality (B.18) still holds. Combining (B.18) and (B.26), we have

\[
\Phi_{k+1} - \Phi_k = \nu (f(X_{k+1}) - f(X_k)) + (1 - \nu) \frac{1}{L^2} \left( \Lambda_{k+1} \| \nabla f(X_{k+1}) \|^2 - \Lambda_k \| \nabla f(X_k) \|^2 \right)
\]

\[
\leq - \nu \eta_1 \kappa_{fcd} \frac{1}{\kappa_{bhm}} \min \left\{ \frac{1}{\Lambda_k}, \| G_k \|^2 + 2(1 - \nu) \gamma \Lambda_k \left( \mu_{\max}^2 \| G_k \|^2 + \frac{1}{L^2} \| \nabla f(X_k) \|^2 \right) \right\}.
\]

Then we choose a suitable \( \nu \in (0, 1) \) such that

\[
\frac{1 - \nu}{\nu} \leq \frac{\eta_1 \kappa_{fcd}}{\kappa_{bhm} \mu_{\max}^2},
\]

which implies that

\[
- \nu \frac{\eta_1 \kappa_{fcd}}{\kappa_{bhm} \mu_{\max}^2} \min \left\{ \frac{1}{\Lambda_k}, \| G_k \|^2 + 2(1 - \nu) \gamma \Lambda_k \right\} \leq 0.
\]

Thus, it follows that

\[
\Phi_{k+1} - \Phi_k \leq 2(1 - \nu) \gamma \frac{1}{L^2} \Lambda_k \| \nabla f(X_k) \|^2.
\]

**(b) Iteration \( k \) is failed.** Here, we have \( X_{k+1} = X_k \) and \( \Lambda_{k+1} = \frac{1}{\gamma} \Lambda_k \). In this case, (B.23) holds.

No matter whether the iteration \( k \) is successful or failed, we always have

\[
\mathbb{E} \left[ \mathbb{I}_{\{1 - I_k \} J_k = 1} (\Phi_{k+1} - \Phi_k) \right] \leq \mathbb{I}_{\{1 - I_k \} J_k = 1} 2(1 - \nu) \gamma \frac{1}{L^2} \Lambda_k \| \nabla f(X_k) \|^2.
\]

Taking conditional expectation on the above inequality, we obtain

\[
\mathbb{E} \left[ \mathbb{I}_{\{1 - I_k \} J_k = 1} (\Phi_{k+1} - \Phi_k) | F_{k-1}^M, F_k^d \right] \leq 2 \mathbb{P}(1 - I_k) J_k = 1 | F_{k-1}^M, F_k^d ) (1 - \nu) \gamma \frac{1}{L^2} \Lambda_k \| \nabla f(X_k) \|^2. \quad (B.27)
\]

**(iii) Estimates \( \{ F^0_k, F^d_k \} \) are loose \( (J_k = 0) \).**

**(a) Iteration \( k \) is successful.** In this case, we have \( X_{k+1} = X_k + D_k \), and \( \Lambda_{k+1} = \gamma \Lambda_k \). Because iteration \( k \) is accepted and the trial step \( D_k \) satisfies Assumption 3.1, we have

\[
F_k^0 - F_k^d \geq \eta_1 \left( M_k(X_k) - M_k(X_{k+1}) \right)
\geq \eta_1 \kappa_{fcd} \| G_k \| \min \left\{ \| G_k \|, \| B_k \| \| \Delta_k \| \right\}
\geq \eta_1 \kappa_{fcd} \min \left\{ \frac{1}{\kappa_{bhm}}, \Lambda_k \right\} \| G_k \|^2. \quad (B.28)
\]
where the last inequality reuses the facts that $\Delta_k = \Lambda_k \|G_k\|$ and $\|B_k\| \leq \kappa_{bhm}$. Applying (B.28), a successful iteration deduces the following bound for the increment of $f$

$$f(X_{k+1}) - f(X_k) = f(X_{k+1}) - F_{k}^{d} + F_{k}^{d} - F_{k}^{o} + F_{k}^{o} - f(X_k) \\
\leq |f(X_{k+1}) - F_{k}^{d}| + F_{k}^{d} - F_{k}^{o} + |F_{k}^{o} - f(X_k)| \\
\leq -\eta \kappa_{f,cd} \min \left\{ \frac{1}{\kappa_{bhm}}, \Lambda_k \right\} \|G_k\|^2 + |f(X_{k+1}) - F_{k}^{d}| + |F_{k}^{o} - f(X_k)|. $$

Then the upper bound for $\Phi_{k+1} - \Phi_k$ can be evaluated as follows

$$\Phi_{k+1} - \Phi_k \\
= \nu(f(X_{k+1}) - f(X_k)) + (1 - \nu) \frac{1}{L^2} (\Lambda_{k+1} \|\nabla f(X_{k+1})\|^2 - \Lambda_k \|\nabla f(X_k)\|^2) \\
\leq \nu(f(X_{k+1}) - f(X_k)) + 2(1 - \nu) \gamma \Lambda_k (\mu_{\max}^2 ||G_k||^2 + \frac{1}{L^2} \|\nabla f(X_k)\|^2) \\
\leq \nu \left( -\eta \kappa_{f,cd} \min \left\{ \frac{1}{\kappa_{bhm}}, \Lambda_k \right\} \|G_k\|^2 + |f(X_{k+1}) - F_{k}^{d}| + |F_{k}^{o} - f(X_k)| \right) \\
+ 2(1 - \nu) \gamma \Lambda_k \left( \mu_{\max}^2 ||G_k||^2 + \frac{1}{L^2} \|\nabla f(X_k)\|^2 \right), \quad (B.29)$$

where the first inequality uses (B.18), which is still true in this setting. We can choose $\nu \in (0, 1)$ to satisfy

$$\frac{1 - \nu}{\nu} \leq \eta \kappa_{f,cd} \min \left\{ \frac{1}{\kappa_{bhm} \mu_{\max}}, 1 \right\}$$

so that

$$-\nu \eta \kappa_{f,cd} \min \left\{ \frac{1}{\kappa_{bhm}}, \Lambda_k \right\} + 2(1 - \nu) \gamma \Lambda_k \mu_{\max}^2 \leq -\frac{1}{2} \nu \eta \kappa_{f,cd} \min \left\{ \frac{1}{\kappa_{bhm}}, \Lambda_k \right\}.$$ 

Then (B.29) can be rewritten as

$$\Phi_{k+1} - \Phi_k \\
\leq -\frac{1}{2} \nu \eta \kappa_{f,cd} \min \left\{ \frac{1}{\kappa_{bhm}}, \Lambda_k \right\} \|G_k\|^2 + 2(1 - \nu) \gamma \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2 \\
+ \nu(|f(X_{k+1}) - F_{k}^{d}| + |F_{k}^{o} - f(X_k)|). \quad (B.30)$$

(b) Iteration $k$ is failed. In this case, we have $X_{k+1} = X_k$ and $\Lambda_{k+1} = \frac{1}{\gamma} \Lambda_k$. Then the inequality (B.23) holds in this setting.

Compared to (B.23), we see that (B.30) dominates the upper bound of $\Phi_{k+1} - \Phi_k$. Taking conditional expectations on (B.30) and applying Assumption 2.1(iii), we have

$$E \left[ \mathbb{1}_{\{(1-J_k)=1\}} (\Phi_{k+1} - \Phi_k) \big| \mathcal{F}_{k-1}^{M,F} \right] \\
\leq \mathbb{P}(1 - J_k = 1 | \mathcal{F}_{k-1}^{M,F}) \left[ -\frac{1}{2} \nu \eta \kappa_{f,cd} \min \left\{ \frac{1}{\kappa_{bhm}}, \Lambda_k \right\} \|G_k\|^2 + \nu(2\kappa_f \Delta_k^2) \right] \\
+ 2 \mathbb{P}(1 - J_k = 1 | \mathcal{F}_{k-1}^{M,F}) (1 - \nu) \gamma \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2 \\
\leq \mathbb{P}(1 - J_k = 1 | \mathcal{F}_{k-1}^{M,F}) \nu \left( -\frac{1}{2} \eta \kappa_{f,cd} \min \left\{ \frac{1}{\kappa_{bhm}}, \Lambda_k \right\} + 2\kappa_f \Lambda_k^2 \right) \|G_k\|^2 \\
+ 2 \mathbb{P}(1 - J_k = 1 | \mathcal{F}_{k-1}^{M,F}) (1 - \nu) \gamma \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2.$$
From the relations that \( \kappa_f \leq \frac{\mu_{\kappa, \text{cd}}}{\mu_{\text{max}}} \min \left\{ \frac{1}{\kappa_{\text{bbr}, \mu_{\text{max}}} \mu_{\text{max}}} \right\} \) and \( \Lambda_k \leq \mu_{\text{max}} \), it follows that

\[
-\frac{1}{2} \eta_1 \mu_{\text{cd}} \min \left\{ \frac{1}{\kappa_{\text{bbr}}}, \Lambda_k \right\} + 2 \kappa_f \Lambda_k^2 \leq 0.
\]

Thus

\[
\mathbb{E} \left[ \|I_{(1-J_k=1)}(\Phi_{k+1} - \Phi_k)\|_{\mathcal{F}_{k-1}^{M,F}} \right] 
\leq 2 \mathbb{P}(1-J_k = 1) \frac{(1-\nu)\gamma \Lambda_k}{L^2} \|\nabla f(X_k)\|^2. \tag{B.31}
\]

Now combining (B.25), (B.27) and (B.31), we can show that

\[
\mathbb{E} \left[ \Phi_{k+1} - \Phi_k \right]_{\mathcal{F}_{k-1}^{M,F}}
= \mathbb{E} \left[ (1-I_k)J_k = 1 \right] + \mathbb{E} \left[ (1-I_k)J_k = 1 \right] + \mathbb{E} \left[ (1-J_k = 1) \right] \frac{1}{\mathcal{F}_{k-1}^{M,F}} \|\nabla f(X_k)\|^2
\leq - \mathbb{P}(I_kJ_k = 1) |\mathcal{F}_{k-1}^{M,F}| (1-\nu) \frac{1}{\gamma} \Lambda_k \|\nabla f(X_k)\|^2
+ 2 \mathbb{P}(1-J_k = 1) \frac{1}{\mathcal{F}_{k-1}^{M,F}} (1-\nu) \gamma \Lambda_k \|\nabla f(X_k)\|^2
\leq - \alpha \beta (1-\nu) \frac{1}{\gamma} \Lambda_k \|\nabla f(X_k)\|^2 + 2(1-\alpha \beta)(1-\nu) \gamma \Lambda_k \|\nabla f(X_k)\|^2
\leq - \frac{1}{2} \alpha \beta (1-\nu) \frac{1}{\gamma} \Lambda_k \|\nabla f(X_k)\|^2. \tag{B.32}
\]

By simply calculating, we have that \( \mathbb{P}(1-I_kJ_k = 1) |\mathcal{F}_{k-1}^{M,F}| \geq \alpha \beta \). The second inequality of (B.32) uses the fact that event \( \{(1-I_k)J_k = 1\} \) and \( \{(1-J_k = 1)\} \) are disjoint, which implies that

\[
\{(1-I_k)J_k = 1\} \cup \{(1-J_k = 1)\} = \{1-I_kJ_k = 1\}.
\]

Thus, we have \( \mathbb{P}(1-I_kJ_k = 1) \cup \{(1-J_k = 1)\} \leq \mathbb{P}(1-I_kJ_k = 1) |\mathcal{F}_{k-1}^{M,F}| \leq 1-\alpha \beta \). Choosing suitable \( \alpha \) and \( \beta \) such that

\[
-\alpha \beta (1-\frac{1}{\gamma}) + 2(1-\alpha \beta) \gamma \leq - \frac{1}{2} \alpha \beta (1-\frac{1}{\gamma}),
\]

which implies that

\[
\alpha \beta \geq \frac{4 \gamma^2}{4 \gamma^2 + (\gamma - 1)} < 1,
\]

then we get the last inequality of (B.32). The proof is completed. \( \square \)
Proof. (of Theorem 3.2) For all $k \leq T$, we know $\|\nabla f(X_k)\| \geq \epsilon$. Let us recall the definition of $\Phi_k$, i.e.

$$\Phi_k = \nu(f(X_k) - f^*) + (1 - \nu) \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2.$$  

In this case, $h(\cdot)$ can be defined as

$$h(\Lambda_k) = \Lambda_k \epsilon^2$$

where $h$ is non-decreasing on any positive domain. From Theorem 3.1, we know that Assumption A.1(3) will hold if the conditions in Theorem 3.1 are satisfied. Applying the result in Lemma B.1, Assumption A.1(2) holds under the conditions that $\alpha \beta > \frac{1}{2}$ and $\hat{\Lambda}$ is defined as (B.10). Clearly Assumption A.1(1) holds. Thus, we can conclude that Assumption A.1 holds under certain conditions. Thus the conclusion of Theorem A.1 is true in this case. Then the following complexity result for Algorithm 3.1 can be achieved by simple analysis. □

Proof. (of Theorem 3.3) In this case, $f$ satisfies Assumption 3.4. Let us consider the stochastic process $\{\Lambda_k, \Psi_k\}$ with

$$\Psi_k = \frac{1}{\nu} - \frac{1}{\Phi_k}.$$  

The convexity of $f$ implies that

$$f(x) - f(y) \geq \nabla f(y)^T (x - y).$$

Let $x = x^*$, $y = X_k$, it follows from the above inequality that

$$f(X_k) - f(x^*) \leq \nabla f(X_k)^T (X_k - x^*) \leq \|\nabla f(X_k)\| \|X_k - x^*\|. \quad (B.33)$$

Because $f$ is $L$-smooth, we have $\|\nabla f(X_k) - \nabla f(x^*)\| \leq L \|X_k - x^*\|$. Due to Assumption 3.4, we know the level set $\mathcal{L}$ is bounded, then

$$\|\nabla f(X_k)\| \leq LD. \quad (B.34)$$

Combining (B.33) and (B.34), we have

$$\Phi_k = \nu(f(X_k) - f^*) + (1 - \nu) \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2$$

$$\leq \nu \|\nabla f(X_k)\| \|X_k - x^*\| + (1 - \nu) \frac{1}{L^2} \mu_{\text{max}} \|\nabla f(X_k)\|^2$$

$$\leq \left( \nu + (1 - \nu) \frac{1}{L} \mu_{\text{max}} \right) D \|\nabla f(X_k)\|.$$  

From the above inequality and the result of Theorem 3.1, we have

$$\mathbb{E} \left[ \Phi_{k+1} - \Phi_k | X_{k-1} \right] \leq -\frac{1}{2} \alpha \beta (1 - \nu) (1 - \frac{1}{\gamma}) \frac{1}{L^2} \Lambda_k \|\nabla f(X_k)\|^2$$

$$\leq -\frac{\alpha \beta (1 - \nu) (1 - \frac{1}{\gamma})}{2(\nu L + (1 - \nu) \mu_{\text{max}})^2 D^2} \Lambda_k \cdot \Phi_k^2. \quad (B.35)$$
The above inequality implies that
\[ E \left[ \Phi_{k+1} | F_{k-1}^{M,F} \right] \leq E \left[ \Phi_k | F_{k-1}^{M,F} \right] = \Phi_k. \]
Recalling the definition of \( \Psi_k \), for all \( k < T \epsilon \), we have
\[ E \left[ \Psi_{k+1} - \Psi_k | F_{k-1}^{M,F} \right] = E \left[ \frac{1}{\Phi_k} - \frac{1}{\Phi_{k+1}} | F_{k-1}^{M,F} \right] \leq - \frac{\alpha \beta (1 - \nu)(1 - \frac{1}{\gamma})}{2(\nu L + (1 - \nu)\mu_{\text{max}})^2 D^2} \cdot \Lambda_k \cdot \Phi_k^2 \Phi_k E \left[ \Phi_{k+1} | F_{k-1}^{M,F} \right] \leq - \alpha \beta (1 - \nu)(1 - \frac{1}{\gamma}) \frac{\Phi_k^2}{2(\nu L + (1 - \nu)\mu_{\text{max}})^2 D^2} \cdot \Lambda_k. \] (B.36)

The first inequality of (B.36) follows from Jensen’s inequality which will be given in Lemma 5. The second inequality uses (B.35). The last inequality is due to the fact that \( E \left[ \Phi_{k+1} | F_{k-1}^{M,F} \right] \leq \Phi_k \). Here, we define an non-decreasing function \( h(\cdot) \) as follows
\[ h(\Lambda_k) = \Lambda_k, \]
and \( C = \frac{\alpha \beta (1 - \nu)(1 - \frac{1}{\gamma})}{2(\nu L + (1 - \nu)\mu_{\text{max}})^2 D^2} \). Then we know that Assumption A.1(3) holds.

From Lemma B.1, we can easily obtain that if \( \alpha \beta > \frac{1}{2} \) and \( \hat{\Lambda} \) is defined as (B.10), Assumption A.1(2) satisfies. Then we can conclude that Assumption A.1 holds. Thus, the conclusion of Theorem A.1 is true.

Finally, substituting the expression of \( \Psi_0, \hat{\Lambda} \) and \( h \) into Theorem A.1, we have
\[ E [T]\epsilon] \leq \frac{\alpha \beta}{2\alpha \beta - 1} \left( \frac{M}{\epsilon} \right) + 1, \]
where \( M = \frac{2(\nu L + (1 - \nu)\mu_{\text{max}})^2 D^2}{\alpha \beta \nu (1 - \nu)(1 - \frac{1}{\gamma}) \Lambda} \). This completes the proof of Theorem 3.3. \( \square \)

Proof. (of Theorem 3.4) In this setting, \( f \) is \( \sigma \)-strongly convex with \( \sigma > 0 \). We will consider the measure function \( \Psi_k \) as follows
\[ \Psi_k = \log(\Phi_k) + \log \left( \frac{1}{\nu \epsilon} \right), \]
to analyse the theoretical complexity. Due to the strongly convexity, we have
\[ f(X_k) - f^* \leq \frac{1}{2\sigma} \| \nabla f(X_k) \|^2. \]
Then
\[ \Phi_k = \nu (f(X_k) - f^*) + (1 - \nu) \frac{1}{L^2} \Lambda_k \| \nabla f(X_k) \|^2 \]
\[ \leq \nu \frac{\| \nabla f(X_k) \|^2}{2\sigma} + (1 - \nu) \frac{1}{L^2} \mu_{\text{max}} \| \nabla f(X_k) \|^2 \]
\[ \leq \left( \frac{\nu}{2\sigma} + (1 - \nu) \frac{1}{L^2} \mu_{\text{max}} \right) \| \nabla f(X_k) \|^2. \] (B.37)
It follows from (B.37) and Theorem 3.1 that
\[
\mathbb{E} \left[ \Phi_{k+1} - \Phi_k \mid F_{k-1}^{M,F} \right] \leq -\frac{1}{2} \alpha \beta (1 - \nu) (1 - \frac{1}{\gamma}) \cdot \frac{\Lambda_k}{L^2} \cdot \| \nabla f(X_k) \|^2
\]
\[
\leq -\frac{\alpha \beta (1 - \nu) (1 - \frac{1}{\gamma})}{\nu L^2 + (1 - \nu) \mu_{\text{max}}} \Lambda_k \Phi_k.
\]
The above inequality implies
\[
\mathbb{E} \left[ \Phi_{k+1} \mid F_{k-1}^{M,F} \right] \leq \left( 1 - \frac{\alpha \beta (1 - \nu) (1 - \frac{1}{\gamma})}{\nu L^2 + (1 - \nu) \mu_{\text{max}}} \Lambda_k \right) \Phi_k.
\]
Recalling the definition \( \Psi_k = \log(\Phi_k) + \log(\frac{1}{\nu \epsilon}) \), we have
\[
\mathbb{E} \left[ \Psi_{k+1} - \Psi_k \mid F_{k-1}^{M,F} \right] = \mathbb{E} \left[ \log(\Phi_{k+1}) - \log(\Phi_k) \mid F_{k-1}^{M,F} \right]
\]
\[
\leq \log \left( \mathbb{E} \left[ \Phi_{k+1} \mid F_{k-1}^{M,F} \right] \right) - \log(\Phi_k)
\]
\[
\leq \log \left( 1 - \frac{\alpha \beta (1 - \nu) (1 - \frac{1}{\gamma})}{\nu L^2 + (1 - \nu) \mu_{\text{max}}} \Lambda_k \right)
\]
\[
\leq -\frac{\alpha \beta (1 - \nu) (1 - \frac{1}{\gamma})}{\nu L^2 + (1 - \nu) \mu_{\text{max}}} \Lambda_k.
\]
In this case, we can define
\[
h(\Lambda_k) = \Lambda_k
\]
and \( C = \alpha \beta (1 - \nu) (1 - \frac{1}{\gamma}) / (\nu L^2 + (1 - \nu) \mu_{\text{max}}) \). From Lemma B.1, we can easily see that Assumption A.1(2) satisfies if \( \alpha \beta > \frac{1}{2} \) and \( \hat{\Lambda} \) is defined as (B.10). Thus we can conclude that Assumption A.1 holds. So the conclusion of Theorem A.1 is true in the strongly convex setting.

By substituting the expression of \( \Psi_0 \), \( \hat{\Lambda} \) and \( h \) into Theorem A.1, we have
\[
\mathbb{E} [T_\epsilon] \leq \frac{\alpha \beta}{2 \alpha \beta - 1} \cdot M \log \left( \frac{1}{\epsilon} \right) + 1,
\]
where \( M = \frac{(\nu L^2 + (1 - \nu) \mu_{\text{max}})}{\alpha \beta (1 - \nu) (1 - \frac{1}{\gamma})} \). Now the proof is finished. \( \square \)

**Appendix C. Related Lemmas and Algorithms for STRME**

**Lemma C.1. (Chebyshev Inequality [43,44])**. If \( X \) is a random variable with mean \( \mathbb{E} [X] \) and variance \( \text{Var}(X) \), then
\[
\mathbb{P}(|X - \mathbb{E} [X]| \geq \nu) \leq \frac{\text{Var}(X)}{\nu^2}, \quad \forall \nu > 0.
\]

The following is the Chebyshev Inequality with conditional expectation. The proof is the same as the unconditional case (see exercise 4.1.2 of [43]).
Lemma C.2. If $X$ is a random variable given the $\sigma$-field $\mathcal{F}$, then
\[
P(\|X - \mathbb{E}[X] \| \geq v \mid \mathcal{F}) \leq \frac{\text{Var}[X \mid \mathcal{F}]}{v^2}, \quad \forall v > 0.
\]

Lemma C.3. (Jensen Inequality [45].) Assume that $f$ is continuous and convex. If $X$ is a random variable, then
\[
\mathbb{E}[f(X)] \geq f(\mathbb{E}[X]).
\]

Remark C.1. If $f$ is concave in Lemma C.3, then we get the opposite result, i.e.
\[
\mathbb{E}[f(X)] \leq f(\mathbb{E}[X]).
\]

Algorithm C.1. STRME with dogleg for the logistic regression problem

1: **Initialization**: initial point $x_0$, $\gamma > 1$, $\eta_1 \in (0, 1)$, $\mu_0$, $SFO_{\text{max}}$, $\epsilon = 10^{-8}$, $\text{TotalSFO}=0$; Set $k = 0$
2: while $\text{TotalSFO} \leq SFO_{\text{max}}$ do
3: Compute $g_k = \frac{1}{p_k} \sum_{i \in O_k} \nabla f_i(x_k) = \frac{1}{p_k} \sum_{i \in O_k} \nabla^2 f_i(x_k)$, where the mini-batch set $O_k$ is randomly chosen
4: $\text{TotalSFO} = \text{TotalSFO} + p_k$
5: Compute $\delta_k = \mu_k \|g_k\|
6: Compute the Cauchy point $d^u = -\frac{g_k^T g_k}{g_k^T B_k g_k} g_k$
7: if $\|d^u\| \geq \delta_k$ then
8: $d_k = d^u$
9: else
10: Compute the Newton step $d^B = -B_k^{-1} g_k$
11: if $\|d^B\| \leq \delta_k$ then
12: $d_k = d^B$
13: else
14: Compute $t_b$ to satisfy $\|d^u + t(d^B - d^u)\| = \delta_k$
15: $d_k = d^u + t_b (d^B - d^u)$
16: end if
17: end if
18: Compute Pred = $-(g_k^T d_k + \frac{1}{2} d_k B_k d_k)$
19: Obtain estimates $f_k^0$ and $f_k^d$ of $f(x_k)$ and $f(x_k + d_k)$
20: if $f_k^0 - f_k^d \geq \eta_1 \text{Pred}$ then
21: $x_{k+1} = x_k + d_k$
22: $\mu_{k+1} = \min(\gamma \mu_k, \mu_{\text{max}})$
23: else
24: $x_{k+1} = x_k$
25: $\mu_{k+1} = \mu_k / \gamma$
26: end if
27: Set $k := k + 1$
28: end while
Algorithm C.2. STRME with L-SR1 for the DNN problem

1: **Initialization:** initial point \( x_0 \), initial trust region radius \( \delta_{-1}, \alpha_2, p_0, p_{\max}, \gamma > 1, \eta_1 \in (0, 1), \mu_0, \mu_{\max}, SFO_{\max} \). TotalSFO = 0, \( S_0 = \phi \) (an empty set); Set \( k = 0 \)

2: **while** TotalSFO \( \leq SFO_{\max} \) **do**

3: Compute \( g_k = \frac{1}{p_k} \sum_{i \in O_k} \nabla f_i(x_k) \), where the mini-batch set \( O_k \) is randomly chosen without replacement and \( p_k = \min \{ p_{\max}, \max \{ \alpha_2 k + p_0, 1/\delta_{k-1}^2 \} \} \)

4: TotalSFO = TotalSFO + \( p_k \)

5: Compute \( \delta_k = \mu_k \| g_k \| \)

6: **if** len(\( S_0 \)) = 0 **then**

7: \( s_k = -\frac{\delta_k}{\| g_k \|} g_k, B_k = B_0 \)

8: **else**

9: \( s_k = -\frac{\delta_k}{\| g_k \|} g_k, B_k = B_0 \)

10: **end if**

11: Compute \( B_k s_k \)

12: Compute Pred = \( - (q_i^T s_k + \frac{1}{2} s_k B_k s_k) \)

13: Obtain estimates \( f_0^k \) and \( f_s^k \) of \( f(x_k) \) and \( f(x_k + s_k) \)

14: Compute \( \bar{g}_k = \frac{1}{p_k} \sum_{i \in O_k} \nabla f_i(x_k + s_k) \), \( y_k = \bar{g}_k - g_k \)

15: TotalSFO = TotalSFO + \( p_k \)

16: **if** \( f_0^k - f_s^k \geq \eta_1 \text{Pred} \) **then**

17: \( x_{k+1} = x_k + s_k \)

18: \( \mu_{k+1} = \min(\gamma \mu_k, \mu_{\max}) \)

19: **else**

20: \( x_{k+1} = x_k \)

21: \( \mu_{k+1} = \mu_k / \gamma \)

22: **end if**

23: **if** \( \| s_k^T (y_k - B_k s_k) \| \geq r \| s_k \| \| y_k - B_k s_k \| \) **then**

24: \( S_{k+1} = [S_k, s_k], Y_{k+1} = [Y_k, y_k] \)

25: **if** len(\( S_{k+1} \)) \( \geq m \) **then**

26: delete \( S_{k+1}[1], Y_{k+1}[1] \)

27: **end if**

28: **end if**

29: Update \( B_{k+1} \) as Section 4.3.2

30: Set \( k := k + 1 \)

31: **end while**

Acknowledgments. The authors are grateful to two anonymous referees for their valuable comments and suggestions. This research is partially supported by the National Natural Science Foundation of China 11331012 and 11688101.

References


