Fast Armijo line search for stochastic gradient descent

Sajad Fathi Hafshejani (sajad.fathihafshejan@uleth.ca)
University of Lethbridge

Daya Gaur
University of Lethbridge

Shahadat Hossain
University of Lethbridge

Robert Benkoczi
University of Lethbridge

Research Article

Keywords: Non-monotone line search method, stochastic gradient descent, Convergence, non-convex problem

Posted Date: January 10th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2285238/v1

Additional Declarations: No competing interests reported.

Fast Armijo line search for stochastic gradient descent

Sajad Fathi Hafshejani[1*], Daya Gaur[1†], Shahadat Hossain[1†] and Robert Benkoczi[1†]

[1*]Department of Math and Computer Science, University of Lethbridge, 4401 University Dr W, Lethbridge, T1K 3M4, AB, Canada.

*Corresponding author(s). E-mail(s): sajad.fathihafshejan@uleth.ca;
Contributing authors: daya.gaur@uleth.ca; shahadat.hossain@uleth.ca; robert.benkoczi@uleth.ca;
[†]These authors contributed equally to this work.

Abstract

We give an improved non-monotone line search algorithm for stochastic gradient descent (SGD) for functions that satisfy interpolation conditions. We establish theoretical convergence guarantees for the algorithm for strongly convex, convex and non-convex functions. We conduct a detailed empirical evaluation to validate the theoretical results.

Keywords: Non-monotone line search method, stochastic gradient descent, Convergence, non-convex problem

1 Introduction

We study the following optimization problem:

$$\min_{x\in\mathbb{R}^d} f(x) = \min_{x\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} f_i(x), \tag{1}$$

where $f_i : \mathbb{R}^d \to \mathbb{R}$ is the loss function for the i-th training sample and the number of samples is n. This minimization problem is central in machine learning. Several iterative approaches for solving (1) are known [1], of those SGD is a popular one, when n is extremely large [2, 3]. SGD uses a random training sample $i_k \in \{1, 2, ..., n\}$ to calculate an iterate of x using the rule:

$$x_{k+1} = x_k - \eta_k \nabla f_{i_k}(x_k), \tag{2}$$

where $\nabla f_{i_k}(x_k)$ is the gradient of the i_k-th component function at x_k, and positive parameter η_k is the learning rate (also known as the step size). Learning rate is an important parameter in SGD [4]. It is usually assumed that $\nabla f_{i_k}(x_k)$ is an unbiased estimate to ∇f. The success of any SGD scheme depends on the search direction and the learning rate. Therefore, we review some seminal work on computing these two parameters.

Equation (2) can be rewritten as:

$$x_{k+1} = x_k - \eta_k D_k^{-1} m_k, \tag{3}$$

where D_k is the preconditioning matrix, m_k is either $\nabla f_{i_k}(x_k)$, or the first moment of the gradient with momentum parameter β_1 or a bias-corrected first moment of the gradient. This transformation gives the search direction and m_k is rotated and scaled by the inverse of preconditioning matrix D_k. Using this reformulation we can recover some of the popular methods as follows. If $D_k^{-1} = I_k$ (identity) and $m_k = \nabla f_{i_k}(x_k)$, then the method of Robbins and Monro [2] follows. Adagrad of Duchi et al. [5] is obtained in this framework with a non-identity preconditioning matrix and when the momentum parameter is zero. For the Adagrad method, $m_k = \nabla f_{i_k}(x_k)$ and the preconditioning matrix is given by:

$$D_k = diag\left(\sqrt{\sum_{i=1}^{k} \nabla f_{i_k}(x_k) \bigodot \nabla f_{i_k}(x_k)}\right) \tag{4}$$

where $\bigodot$ is the component-wise product of two vectors. The RMSProp scheme of Tieleman and Hinton [6] uses the following parameters:

$$D_k = \sqrt{\beta_2 D_k^2 + (1 - \beta_2) diag\left(\nabla f_{i_k}(x_k) \bigodot \nabla f_{i_k}(x_k)\right)} \tag{5}$$

and $m_k = \nabla f_{i_k}(x_k)$.

The convergence of SGD is mainly affected by the learning rate. The convergence analysis of SGD with constant step size for convex and strongly convex functions was performed in [7–9]. Convergence for the case when the learning rate is decreased over iterations was shown in [2, 10, 11]. Loizou et al. [12], showed that an improved Polyak scheme

$$\eta_k = \min\left\{\frac{f_{i_k}(x_k) - f_{i_k}^*}{\|c\nabla f_{i_k}(x_k)\|^2 + \delta}, \omega_b\right\}, \tag{6}$$

where $\omega_b > 0$ is a constant and $c \in \mathbb{R}_+$ converges for strong convex, convex, and non-convex functions. Tan et al. [13] used the Barzilai-Borwein (BB) step size, given by,

$$\eta_k = \frac{1}{m} \frac{\|x_k - x_{k-1}\|^2}{(x_k - x_{k-1})^T (\nabla f(x_k) - \nabla f(x_{k-1}))},$$

where m in the number of inner iterations and $\nabla f(x_k)$ is the full gradient at point x_k, in SGD and demonstrated that their approach has good performance in practice. Yang [14] proposed a new effective BB learning rate for a variance-reduced algorithm for a non-convex objective. Armijo line search is another very successful method to determine the learning rate in each iteration [1]. The Armijo condition used to determine the learning rate is

$$f_{i_k}(x_k - \eta_k \nabla f_{i_k}(x_k)) \leq f_{i_k}(x_k) - c\eta_k \|\nabla f_{i_k}(x_k)\|^2 \tag{7}$$

where $c \in (0, 1)$. Vaswani et al. [15] were the first to propose the use of the Armijo line search technique in SGD. They proved that SGD with a stochastic variant of the classic Armijo line-search has deterministic convergence for convex and strongly convex functions. Recently, Vaswani et al. [16] have studied convergence of stochastic linear search for SGD which is adaptive to noise in the stochastic gradients.

1.1 Contributions

The main contribution of this paper is a non-monotone technique to determine the learning rate for SGD and its convergence analysis. We extend the proof techniques of Vaswani et al. [15] to our method (Algorithm 1) and prove strong convergence guarantees for SGD for strongly convex-functions in Theorem 1, convex functions in Theorem 2 and non-convex functions in Theorem 3.

We perform a comparative experimental analysis of our methods against the popular methods for determining search direction SGD [2], Adagrad [5], RMSProp [6] and methods for determining the learning rate due to [12, 15]. Our experiments conclusively show that Algorithm 1 has faster convergence on the data sets tested. Figure 1 shows the behaviour of the loss functions for the Armijo line search and non-monotone line search algorithms on two datasets in the first 1000 iterations. The non-monotone line search algorithm achieves a better convergence than the Armijo line search algorithm. For the loss function to get to a specific ϵ, the non-monotone line search algorithm needs fewer iterations. Moreover, this figure demonstrates the efficiency of the non-monotone line search algorithm to the Armijo line search method in [15].

The rest of this paper is organized as follows. Section 2 describes non-monotone line search for SGD. Section 3 lists the convergence results for the proposed algorithm for strong convex, convex, and non-convex functions. Section 4 is on empirical validation of our technique against well-known methods of [2, 5, 6, 12, 15]. The novelty of the results here is further addressed in

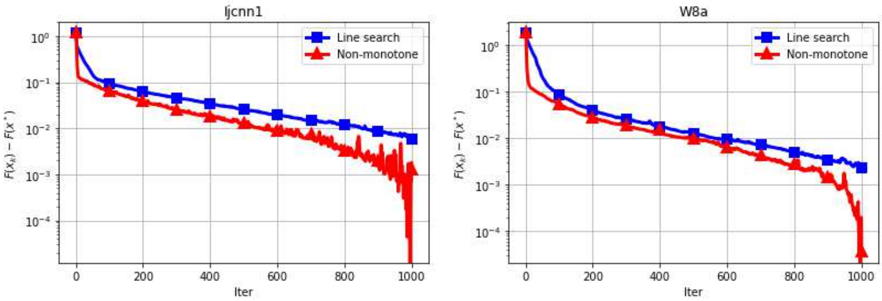

Fig. 1 The values of losses for solving logistic regression problem based on Non-monotone line search and Armijo line search on two different data-sets.

the Section 5. We end with remarks and some open questions in Section 6. All the proofs, for the convergence results stated in Section 3, are in the appendix.

2 Non-monotone line search

We first describe the non-monotone line search approach of Grippo et al. [17]. We state the SGD algorithm of [2]. We then introduce Algorithm 1 which is non-monotonic stochastic gradient descent.

The non-monotone line search [17, 18] computes the learning rate in each iteration using the following equations:

$$f(x_k - \eta_k \nabla f(x_k)) \leq f_{l(k)} - c\eta_k \|\nabla f(x_k)\|^2, \tag{8}$$

where $c \in (0, 1)$

$$f_{l(k)} = \max_{0 \leq j \leq m(k)} f(x_{k-j}), \qquad k = 0, 1, 2, ... \tag{9}$$

and $m(0) = 0,\ 0 \leq m(k) \leq \min\{m(k-1) + 1, N\}$ for all $k \geq 1$ and $N \geq 0$.

An efficient version of the non-monotone line search method was given by Ahookhosh et al. [19] using convex combination of $f_{l(k)}$ and $f(x_k)$ as follows

$$f(x_k - \eta_k \nabla f(x_k)) \leq R(x_k) - c\eta_k \|\nabla f(x_k)\|^2, \tag{10}$$

where $c \in (0, 1)$, and

$$R(x_k) = \gamma_k f_{l(k)} + (1 - \gamma_k) f(x_k),$$

for $m(0) = 0,\ 0 \leq m(k) \leq \min\{m(k-1) + 1, N\}$ for all $k \geq 1$ and $N \geq 0$.

Assume that the current iteration of the algorithm is k. Algorithm 1 constructs a new point as follows:

$$x_{k+1} = x_k - \eta_k \nabla f_{i_k}(x_k)$$

The learning rate is computed using the following equation:

$$f_{i_k}(x_k - \eta_k \nabla f_{i_k}(x_k)) \le R_{i_k}(x_k) - c\eta_k \|\nabla f_{i_k}(x_k)\|^2, \tag{11}$$

where $c \in (0,1)$, and $f_{i_k l(k)}$ and $R_{i_k}(x_k)$ are defined as:

$$R_{i_k}(x_k) = \gamma_k f_{i_k l(k)} + (1-\gamma_k) f_{i_k}(x_k),$$

and

$$f_{i_k l(k)} = \max_{0 \le j \le m(k)} f_{i_k}(x_{k-j}), \qquad k = 0, 1, 2, ... \tag{12}$$

Determining the value of the non-monotone parameter γ is the next important issue. Ahookhosh et al. [19] proposed an adaptive method to compute γ using the recurrence $\gamma_k = \frac{1}{3}\gamma_0(-\frac{1}{2})^k + \frac{2}{3}\gamma_0$ where $\gamma_0 = 0.15$ and $\lim_{k\to\infty} \gamma_k = 0.1$. In this scheme γ_k is close to 0.1 after only a few iterations and γ_k is independent of the objective value.

The line search method for SGD proposed by [15] may not evaluate an effective value for the step size when the norm of gradient is too large. In order to better understand, we can imagine that the norm of gradient is large. In this case, the step size generated by Armijo line search algorithm [15] is extremely tiny, the algorithm needs a large number of iterations.

In the non-monotone strategy some growth in the function value is permitted, which leads the non-monotone line search to choose a larger step size at each iteration than the Armijo line search method. This will also dramatically decrease the number of iterations of the algorithm than the Armijo line search proposed in [15]. The existing schemes for determining the parameter γ_k work based on the number of iterations [19] or constant values, and may not reduce the value of the objective function significantly in initial iterations. To overcome this drawback, we propose a new scheme for choosing γ_k based on the gradient behaviour of the objective function. This can reduce the total number of iteration.

We propose an adaptive strategy to compute γ_k, needed to the convex-combination, as follows:

$$\gamma_k = 1 - \omega \exp(-\|\nabla f_{i_k}(x_k)\|), \tag{13}$$

where $\omega \in (0,1]$.

3 The Non-monotone Line Search Algorithm

We explain the algorithm (pseudo-code in Algorithm 1) for the case when the search direction d_k is given by the negative gradient (on line 5). The algorithm starts with the initial point x_0 and specific values of the parameters. Number i_k is randomly chosen at line 4. Then the search direction d_k is calculated on line 5. The algorithm can also use any of equations (4), (5). Next on line 6 we have the initial value of the learning rate. Next on line 7, using the initial value of learning rate and the search direction, a new point is computed using equation (2). After computing the new point, the algorithm updates the non-monotone parameter γ_k using (13) on line 8. On line 9, we recompute the value of R_{i_k}. Lines 10-13 are used to compute η and the new point until the condition in Eq. (11) is met. On line 14, the algorithm assigns a new value to the learning rate η_k. In line 15, we compute the new point and the process continues for T iterations. If the algorithm uses a search direction given by either Equation (4) or (5) then line 5 needs to be changed accordingly.

The non-monotone line search determines a more appropriate step size than the Armijo line search method when the gradient norm is significant, as in the case the sequence of iterations follows the bottom of a narrow valley. In fact, from step 10 of Algorithm1, it is evident that the non-monotone line search method can allow the algorithm to use a more significant step size than the Armijo line search method. Several works show that the non-monotone line search method is faster than the Armijo line search method [19, 20].

Algorithm 1 Non-monotone Line Search for SGD

```
 1: INPUT: $T, x_0,\ c \in (0,1), N = 5,\ \rho = 0.25,\ \eta_0 = \eta_{max} = 1, \omega \in (0,1]$
 2: OUTPUT: $x^*$
 3: FOR $k \in [1 \dots T]$
 4:     $i_k \leftarrow random([1 \dots n])$
 5:     $d_k \leftarrow -\nabla f_{i_k}$                                   ▷ search direction
 6:     $\eta \leftarrow \eta_0$
 7:     $x_{k+1} \leftarrow x_k + \eta d_k$
 8:     $\gamma_k \leftarrow 1 - \omega \exp(-\nabla f_{i_k})$                    ▷ Eq. 13
 9:     $R_{i_k} \leftarrow \gamma_k f_{i_k l(k)} + (1 - \gamma_k) f_{i_k}(x_k)$
10:     WHILE $(f_{i_k}(x_{k+1}) \geq R_{i_k} + \eta c \nabla f_{i_k}^T d_k)$       ▷ Eq. 11
11:         $\eta \leftarrow \eta \rho$
12:         $x_{k+1} \leftarrow x_k + \eta d_k$
13:     END
14:     $\eta_k \leftarrow \eta$
15:     $x_{k+1} \leftarrow x_k + \eta_k d_k$                                ▷ Eq. 2
16: END
```

We describe the difference between Algorithm 1 and the algorithms in [15, 19]. For one, Algorithm 1 uses a non-monotone line search method to calculate the step size value in each iteration (line 10). The algorithm in [15] uses a line search approach to find the best value for the step size. Therefore, Algorithm 1 differs from the algorithm in [15] in the method use to calculate the step size. This paper's proposed non-monotone line search method uses an adaptive approach for calculating the non-monotone parameter, which differs from Algorithm 1 in [19] where the method is non-adaptive.

3.1 Assumptions

We use the following assumptions. All the convergence results rely on them.

- A1: We consider the problem of minimizing a continuous function $f : \mathbb{R}^d \to \mathbb{R}$. The function f has a finite-sum structure, meaning that $f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$ where n is the number of points in the training set. Function f_i is the loss function for the training point i. Depending on the model, f can be strongly-convex, convex, or non-convex.
- A2: f is bounded below, that is, there exists at least one $x^* \in \mathbb{R}^d$, such that:

$$f(x) \geq f(x^*) \quad \forall\, x \in \mathbb{R}^d$$

- A3: f is differentiable and L-smooth, meaning the mapping $x \to \nabla f(x)$ is an L-Lipschitz function, that is:

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

- A4: If function f is minimized at x^* that is $\nabla f(x^*) = 0$, then for all loss functions f_i we have that $\nabla f_i(x^*) = 0$.
- A5: The function f satisfies minimizer interpolation, that is, if for all i and $x \in \mathbb{R}^d$

$$f(x^*) \leq f(x) \Rightarrow f_i(x^*) \leq f_i(x) \quad \text{where } x^* \text{ is the minimizer.}$$

Assumptions A4, A5 together are known as the interpolation conditions. We use the following notation in the rest of the paper:

$$R_{i_k} := R_{i_k}(x_k), \quad f_{i_k} := f_{i_k}(x_k), \quad f_{i_k l(k)} = f_{i_k l}(x_k).$$

Remark 1 The function f is μ-strong convex, i.e., there exists a constant $\mu > 0$ such that $\forall x, x' \in \mathbb{R}^d$

$$f(x) \geq f(x') + \nabla f(x')^T(x - x') + \frac{\mu}{2}\|x - x'\|^2 \tag{14}$$

Remark 2 (Convexity) Let the function $f : \mathbb{R}^d \to \mathbb{R}$ be differentiable. Then f is convex, if and only if for every $x, y \in \mathbb{R}^d$ the inequality

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

is satisfied.

Remark 3 (co-coercivity). If $f(x) : \mathbb{R}^d \to \mathbb{R}$ is convex and its gradient is L-Lipschitz continuous, then, we have

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

The next goal is to calculate an upper bound on the step size during each iteration.

Lemma 1 Suppose that the sequence $\{x_k\}$ is generated by Algorithm 1. Then, $f_{i_k l(k)}$ is a decreasing sequence and for all $k \in N \cup \{0\}$.

Proof Using definition R_{i_k}, we have:

$$R_{i_k} = \gamma_k f_{i_k l(k)} + (1-\gamma_k) f_{i_k} \leq \gamma_k f_{i_k l(k)} + (1-\gamma_k) f_{i_k l(k)} = f_{i_k l(k)}. \tag{15}$$

It follows that:

$$f_{i_k}(x_{k+1}) \leq R_{i_k} - c\eta_k \|\nabla f_{i_k}\|^2 \leq f_{i_k l(k)} - c\eta_k \|\nabla f_{i_k}\|^2.$$

That means:

$$f_{i_k}(x_{k+1}) \leq f_{i_k l(k)}. \tag{16}$$

On the other hand, using (12), we show that $f_{i_k l(k+1)} \leq f_{i_k l(k)}$. To this end, we have:

$$\begin{aligned} f_{i_k l(k+1)} &= \max_{0 \leq j \leq m(k+1)} \left\{ f_{i_k}(x_{k+1-j}) \right\} \\ &\leq \max \left\{ \max_{0 \leq j \leq m(k)} \left\{ f_{i_k}(x_{k-j}) \right\}, f_{i_k}(x_{k+1}) \right\} \\ &= \max \left\{ f_{i_k l(k)}, f_{i_k}(x_{k+1}) \right\} = f_{i_k l(k)}, \end{aligned}$$

which shows that the sequence $f_{i_k l(k)}$ is monotonically nonincreasing. Since $f_{i_k l(0)} = f_{i_k}(x_0)$, we deduce $f_{i_k}(x_k) \leq f_{i_k l(k-1)} \leq ... \leq f_{i_k l(0)} = f_{i_k}(x_0)$. □

Lemma 2 Suppose that the sequence $\{x_k\}$ is generated by Algorithm 1. Then we have

$$f_{i_k}(x_k) \leq R_{i_k}(x_k), \qquad \forall\, k \in N \cup \{0\}. \tag{17}$$

Proof From the definition of $f_{i_k l(k)}$, we have $f_{i_k}(x_k) \leq f_{i_k l(k)}$, for all $k \in N \cup \{0\}$. Therefore, we have:

$$\begin{aligned} f_{i_k}(x_k) &= \gamma_k f_{i_k}(x_k) + (1-\gamma_k) f_{i_k}(x_k) \\ &\leq \gamma_k f_{i_k l(k)} + (1-\gamma_k) f_{i_k}(x_k) = R_{i_k}(x_k) \quad \forall\, k \in N \cup \{0\}. \end{aligned}$$

□

Lemma 3 Suppose that the sequence $\{x_k\}$ is generated by Algorithm 1. Then we have:

$$\lim_{k \to \infty} f_{i_k l(k)} = \lim_{k \to \infty} f_{i_k}(x_k).$$

Proof The proof is similar to Lemma 3.2 in [19]. Therefore, we omit it here. □

Lemma 4 For the sequence $\{x_k\}$ generated by Algorithm 1, we have:

$$\lim_{k \to \infty} R_{i_k} = \lim_{k \to \infty} f_{i_k}. \tag{18}$$

Proof From Lemma 2 and (15), we have:

$$f_{i_k}(x_k) \leq R_{i_k}(x_k) \leq f_{i_k l(k)}.$$

The result is obtained by using Squeeze Theorem [21] and Lemma 3. □

Lemma 5 Suppose that x_k is not a stationary point of the algorithm and L_{i_k} denotes the Lipschitz constant of $\nabla f_{i_k}(x_k)$ and f_i's are L-smooth functions. Then a lower bound for $\eta_k \in (0, \eta_{max}]$ is given by:

$$\eta_k \geq \min\left\{\eta_{max}, \frac{2(1-c)}{L_{i_k}}\right\}, \tag{19}$$

Proof From the smoothness of f_{i_k} and the update rule, the following inequality holds for all values of η_k:

$$f_{i_k}(x_{k+1}) \leq f_{i_k}(x_k) - \eta_k(1 - \frac{L_{i_k}\eta_k}{2})\|\nabla f_{i_k}(x_k)\|^2 \tag{20}$$

Using Lemma 2, that is $f_{i_k}(x_k) \leq R_{i_k}(x_k)$. Then, the inequality (20) can be rewritten as:

$$f_{i_k}(x_{k+1}) \leq R_{i_k}(x_k) - \eta_k(1 - \frac{L_{i_k}\eta_k}{2})\|\nabla f_{i_k}(x_k)\|^2 \tag{21}$$

The non-monotone line-search finds a step size satisfying the following condition:

$$f_{i_k}(x_{k+1}) \leq R_{i_k}(x_k) - c\eta_k\|\nabla f_{i_k}(x_k)\|^2 \tag{22}$$

Using the inequalities (21) and (22), we conclude that:

$$\begin{aligned} c\eta_k &\geq \eta_k(1 - \frac{L_{i_k}\eta_k}{2}) \\ \Rightarrow \eta_k &\geq \frac{2(1-c)}{L_{i_k}} \end{aligned}$$

□

Lemma 6 Suppose that the functions f and f_i are convex and f is a $\mu-$strong and interpolation condition is true. Then, we have:

$$\eta_k \leq \frac{1}{2\mu_{i_k}c}.$$

Proof The non-monotone line search condition implies that:

$$\begin{aligned} f_{i_k}(x_{k+1}) &\leq R_{i_k} - c\eta_k\|\nabla f_{i_k}\|^2 \qquad (23) \\ \Rightarrow \eta_k \leq \frac{R_{i_k} - f_{i_k}(x_{k+1})}{c\|\nabla f_{i_k}\|^2} &= \frac{R_{i_k} - f_{i_k}(x^*) + f_{i_k}(x^*) - f_{i_k}(x_{k+1})}{c\|\nabla f_{i_k}\|^2}. \end{aligned}$$

Using interpolation condition, that is, $f_{i_k}(x^*) \leq f_{i_k}(x)$, we conclude that $f_{i_k}(x^*) - f_{i_k}(x_{k+1})$ is negative, so we have

$$\eta_k \leq \frac{R_{i_k} - f_{i_k}(x^*)}{c\|\nabla f_{i_k}\|^2}. \tag{24}$$

On the other hand, if each f_{i_k} satisfies the either strong-convexity or the Polyak-Lojasiewicz (PL) inequality [11, 22] (which is weaker than strong-convexity and does not require convexity), then we have:

$$f_{i_k} - f_{i_k}(x^*) \leq \frac{1}{2\mu_{i_k}} \|\nabla f_{i_k}(x)\|^2. \tag{25}$$

By using Lemma 4, (25), and (24), we conclude that:

$$\eta_k \leq \frac{f_{i_k}(x_k) - f_{i_k}(x^*)}{c\|\nabla f_{i_k}\|^2} \leq \frac{1}{2\mu_{i_k} c}. \tag{26}$$

Therefore, we have:

$$\eta_k \leq \min\left\{\frac{1}{2\mu_{i_k} c}, \eta_{max}\right\}$$

□

Using Lemmas 5 and 6, we can conclude that:

$$\eta_k \in \left[\min\left\{\frac{2(1-c)}{L_{i_k}}, \eta_{max}\right\}, \min\left\{\frac{1}{2\mu_{i_k} c}, \eta_{max}\right\}\right] \tag{27}$$

4 Convergence Results

4.1 Strong-Convex Case

Theorem 1 *(Strongly-Convex) Suppose that f and f_i are smooth functions and f_i are convex and f is a μ-strong convex and satisfies the interpolation condition. Then, the SGD with non-monotone line search with $c = \frac{1}{2}$ in (11) achieves the rate:*

$$\mathbb{E}\left[\|x_T - x^*\|^2\right] \leq \max\left\{(1 - \frac{\bar{\mu}}{L_{max}})^T, (1 - \bar{\mu}\eta_{max})^T\right\} \|x_0 - x^*\|^2$$

where $\bar{\mu} = \sum_{i=1}^{n} \frac{\mu_i}{n}$ and $L_{max} = \max\{L_i\}$ denotes the maximum smoothness constant for all functions f_i.

Proof We evaluate the value of $\|x_{k+1} - x^*\|^2$:

$$\begin{aligned}\|x_{k+1} - x^*\|^2 &= \|x_k - \eta_k \nabla f_{i_k} - x^*\|^2 = \|x_k - x^* - \eta_k \nabla f_{i_k}\|^2 \\ &= \|x_k - x^*\|^2 + \eta_k^2 \|\nabla f_{i_k}\|^2 - 2\eta_k \langle \nabla f_{i_k}, x_k - x^*\rangle\end{aligned}$$

Note that Remark 1 follows that:

$$-\langle \nabla f_{i_k}, x_k - x^*\rangle \leq f_{i_k}(x^*) - f_{i_k}(x_k) - \frac{\mu_{i_k}}{2}\|x_k - x^*\|^2.$$

Therefore, we have:

$$\begin{aligned}\|x_{k+1} - x^*\|^2 &\leq \|x_k - x^*\|^2 + \eta_k^2\|\nabla f_{i_k}\|^2 + 2\eta_k\left[f_{i_k}(x^*) - f_{i_k}(x_k) - \frac{\mu_{i_k}}{2}\|x_k - x^*\|^2\right] \\ &= (1 - \mu\eta_k)\|x_k - x^*\|^2 + 2\eta_k\left[f_{i_k}(x^*) - f_{i_k}(x_k)\right] + \eta_k^2\|\nabla f_{i_k}\|^2\end{aligned}$$

The non-monotone line search condition implies that $\eta_k^2\|\nabla f_{i_k}\|^2 \leq \frac{\eta_k}{c}[R_{i_k} - f_{i_k}(x_{k+1})]$. Therefore, we have:

$$\|x_{k+1} - x^*\|^2 \leq (1 - \mu_{i_k}\eta_k)\|x_k - x^*\|^2 + 2\eta_k[f_{i_k}(x^*) - f_{i_k}(x_k)] + \frac{\eta_k}{c}[R_{i_k} - f_{i_k}(x_{k+1})]$$

$$\leq (1-\mu_{i_k}\eta_k)\|x_k - x^*\|^2 + 2\eta_k[f_{i_k}(x^*) - f_{i_k}(x_k)] + \frac{\eta_k}{c}[R_{i_k} - f_{i_k}(x^*))]$$

Using Lemma 4, we conclude that:

$$\|x_{k+1} - x^*\|^2 \leq (1-\mu_{i_k}\eta_k)\|x_k - x^*\|^2 + (2\eta_k - \frac{\eta_k}{c})[f_{i_k}(x^*) - f_{i_k}(x_k)]$$

The rest of the proof is similar to the proof of Theorem 1 in [15]. We write it here for completeness.

By using the interpolation condition, we can conclude that $f_{i_k}(x^*) - f_{i_k}(x_k)$ is negative. If $c \geq \frac{1}{2}$, then:

$$\|x_{k+1} - x^*\|^2 \leq (1-\mu_{i_k}\eta_k)\|x_k - x^*\|^2$$

Taking expectation wrt to i_k

$$\begin{aligned}\mathbb{E}\left[\|x_{k+1} - x^*\|^2\right] &\leq \mathbb{E}_{i_k}(1-\mu_{i_k}\eta_k)\|x_k - x^*\|^2 \\ &\leq \left(1 - \mathbb{E}_{i_k}\left[\mu_{i_k}\min\left\{\frac{2(1-c)}{L_{i_k}}, \eta_{max}\right\}\right]\right)\|x_k - x^*\|^2\end{aligned}$$

Setting $c = \frac{1}{2}$, we have:

$$\mathbb{E}\left[\|x_{k+1} - x^*\|^2\right] \leq \left(1 - \mathbb{E}_{i_k}\left[\mu_{i_k}\min\left\{\frac{1}{L_{i_k}}, \eta_{max}\right\}\right]\right)\|x_k - x^*\|^2$$

Now, we consider two cases for η_k, that is,:

- If $\eta_{max} < \frac{1}{L_{max}}$. In this case, we can conclude that: $\eta_{max} < \frac{1}{L_{i_k}}$. So, we have:

  $$\begin{aligned}\mathbb{E}\left[\|x_{k+1} - x^*\|^2\right] &\leq (1 - \mathbb{E}_{i_k}[\mu_{i_k}\eta_k])\|x_k - x^*\|^2 \\ &\leq (1 - \mathbb{E}_{i_k}[\mu_{i_k}]\eta_{max})\|x_k - x^*\|^2 = (1-\bar{\mu}\eta_{\max})\|x_k - x^*\|^2\end{aligned}$$

  By recursion through iteration $k = 1$ to T, we have:

  $$\mathbb{E}\left[\|x_T - x^*\|^2\right] \leq (1-\bar{\mu}\eta_{\max})^T\|x_0 - x^*\|^2$$

- When $\eta_{max} \geq \frac{1}{L_{max}}$. In this case, we use this fact $\min\left\{\frac{1}{L_{i_k}}, \eta_{max}\right\} \geq \min\left\{\frac{1}{L_{max}}, \eta_{max}\right\}$. So, we have:

  $$\begin{aligned}\mathbb{E}\left[\|x_{k+1} - x^*\|^2\right] &\leq \left(1 - \mathbb{E}_{i_k}\left[\mu_{i_k}\min\left\{\frac{1}{L_{max}}, \eta_{max}\right\}\right]\right)\|x_k - x^*\|^2 \\ &= \left(1 - \mathbb{E}_{i_k}\left[\mu_{i_k}\frac{1}{L_{max}}\right]\right)\|x_k - x^*\|^2 \\ &= \left(1 - \frac{\mathbb{E}_{i_k}[\mu_{i_k}]}{L_{max}}\right)\|x_k - x^*\|^2 = \left(1 - \frac{\bar{\mu}}{L_{max}}\right)\|x_k - x^*\|^2\end{aligned}$$

  By recursion through iteration $k = 1$ to T, we have:

  $$\mathbb{E}\left[\|x_T - x^*\|^2\right] \leq \left(1 - \frac{\bar{\mu}}{L_{max}}\right)^T \|x_0 - x^*\|^2.$$

Putting the two cases together, we conclude that:

$$\mathbb{E}\left[\|x_T - x^*\|^2\right] \le \max\left\{\left(1 - \frac{\bar{\mu}}{L_{max}}\right)^T, (1 - \bar{\mu}\eta_{\max})^T\right\} \|x_0 - x^*\|^2$$

□

Remark 4 The number of iterations required to reach an ϵ-neighbourhood of x^* is given by:

$$T \ge \max\left\{\frac{\log\epsilon}{\log\left(1 - \frac{\bar{\mu}}{L_{max}}\right)}, \frac{\log\epsilon}{\log(1 - \bar{\mu}\eta_{\max})}\right\}.$$

4.2 Convex Case

Theorem 2 *(Convex) Suppose that f and f_i's are smooth and convex functions, and satisfy the interpolation condition. Then, the SGD with non-monotone line search with $c > \frac{1}{2}$ in (11) achieves the rate:*

$$\mathbb{E}\left[f(\bar{x}_T) - f(x^*)\right] \le \frac{c\max\left\{\frac{L_{max}}{2(1-c)}, \frac{1}{\eta_{max}}\right\}}{(2c-1)T}\|x_0 - x^*\|^2$$

where $\bar{x} = \sum_{i=1}^{n} \frac{x_i}{n}$ denotes the averaged iterate after T iterations and $L_{max} =$ $\max\{L_{i_k}\}$.

Proof We Have:

$$\begin{aligned}\|x_{k+1} - x^*\|^2 &= \|x_k - \eta_k \nabla f_{i_k} - x^*\|^2 = \|x_k - x^* - \eta_k \nabla f_{i_k}\|^2 \\ &= \|x_k - x^*\|^2 - 2\eta_k \langle \nabla f_{i_k}, x_k - x^*\rangle + \eta_k^2 \|\nabla f_{i_k}\|^2.\end{aligned}$$

Therefore, we have:

$$2\eta_k\langle\nabla f_{i_k}, x_k - x^*\rangle = \|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2 + \eta_k^2\|\nabla f_{i_k}\|^2$$

It follows that:

$$\begin{aligned}\langle\nabla f_{i_k}, x_k - x^*\rangle &= \frac{1}{2\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right] + \frac{\eta_k}{2}\|\nabla f_{i_k}\|^2 \\ &\le \frac{1}{2\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right] + \frac{R_{i_k} - f_{i_k}(x_{k+1})}{2c} \\ &\le \frac{1}{2\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right] + \frac{R_{i_k} - f_{i_k}(x^*)}{2c},\end{aligned}$$

where the first inequality is obtained by using non-monotone line search condition and the second inequality is obtained by using interpolation condition. Using Lemma 4, we have:

$$\langle\nabla f_{i_k}, x_k - x^*\rangle \le \frac{1}{2\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right] + \frac{f_{i_k}(x_k) - f_{i_k}(x^*)}{2c}$$

Taking expectation wrt i_k,

$$\mathbb{E}\langle\nabla f_{i_k}, x_k - x^*\rangle \le \mathbb{E}\left[\frac{1}{2\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right]\right] + \mathbb{E}\left[\frac{f_{i_k}(x_k) - f_{i_k}(x^*)}{2c}\right]$$

$$= \mathbb{E}\left[\frac{1}{2\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right]\right] + \left[\frac{f(x_k) - f(x^*)}{2c}\right]$$

We have:

$$\langle \mathbb{E}\nabla f_{i_k}, x_k - x^*\rangle \le \mathbb{E}\left[\frac{1}{2\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right]\right] + \left[\frac{f(x_k) - f(x^*)}{2c}\right]$$

It follows that:

$$\langle \nabla f(x_k), x_k - x^*\rangle \le \mathbb{E}\left[\frac{1}{2\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right]\right] + \left[\frac{f(x_k) - f(x^*)}{2c}\right]$$

Using the convexity property of the function f (see Remark 2) and we can conclude that:

$$\begin{aligned} f(x_k) - f(x^*) \; &\le \; \langle \nabla f(x_k), x_k - x^*\rangle \\ &\le \mathbb{E}\left[\frac{1}{2\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right]\right] + \left[\frac{f(x_k) - f(x^*)}{2c}\right] \end{aligned}$$

If $c \ge \frac{1}{2}$, we have:

$$f(x_k) - f(x^*) \le \mathbb{E}\left[\frac{c}{(2c-1)\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right]\right].$$

Taking expectation and summing from $k = 0$ to $k = T-1$,

$$\mathbb{E}\left[\sum_{k=0}^{T-1}(f(x_k) - f(x^*))\right] \le \sum_{k=0}^{T-1}\left(\frac{c}{(2c-1)\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right]\right)$$

Next we use Jensen's inequality stated below:

$$\mathbb{E}\left[f(\bar{x}_T) - f(x^*)\right] \le \mathbb{E}\left[\sum_{k=0}^{T-1}(\frac{f(x_k) - f(x^*)}{T})\right]$$

in which $\bar{x}_T = \frac{\sum_{i=1}^{T} x_i}{T}$. Thus,

$$\mathbb{E}\left[f(\bar{x}_T) - f(x^*)\right] \le \frac{1}{T}\mathbb{E}\left[\sum_{k=0}^{T-1}(\frac{c}{(2c-1)\eta_k}\left[\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right])\right]$$

If $\Delta_k = \|x_k - x^*\|^2$ then:

$$\mathbb{E}[f(\bar{x}_T) - f(x^*)] \; \le \; \frac{c}{T(2c-1)}\mathbb{E}\left[\sum_{k=0}^{T-1}(\frac{1}{\eta_k}[\Delta_k - \Delta_{k+1}])\right] \tag{28}$$

Using the lower bound for η_k, and we have:

$$\frac{1}{\eta_k} \le \max\left\{\frac{L_{i_k}}{2(1-c)}, \frac{1}{\eta_{max}}\right\} \le \max\left\{\frac{L_{max}}{2(1-c)}, \frac{1}{\eta_{max}}\right\}$$

Therefore, we can rewrite (28) as:

$$\begin{aligned} \mathbb{E}[f(\bar{x}_T) - f(x^*)] \; &\le \; \frac{c\max\left\{\frac{L_{max}}{2(1-c)}, \frac{1}{\eta_{max}}\right\}}{T(2c-1)}\mathbb{E}\left[\sum_{k=0}^{T-1}[\Delta_k - \Delta_{k+1}]\right] \\ &= \; \frac{c\max\left\{\frac{L_{max}}{2(1-c)}, \frac{1}{\eta_{max}}\right\}}{T(2c-1)}\mathbb{E}\left[\Delta_0 - \Delta_T\right] \end{aligned}$$

It implies that:

$$\mathbb{E}[f(\bar{x}_T) - f(x^*)] \le \frac{c\max\left\{\frac{L_{max}}{2(1-c)}, \frac{1}{\eta_{max}}\right\}}{T(2c-1)}\|x_0 - x^*\|^2$$

□

4.3 Non-Convex case

We state the convergence results for the non-convex objective function.

We assume that the function f satisfies the strong growth condition (SGC) with constant κ, that is $\mathbb{E}[\|\nabla f_{i_k}(x)\|^2] \leq \kappa \|\nabla f(x)\|^2$ holds for any point x_k [15]. This assumption implies that if $\nabla f(x) = 0$, then $\nabla f_{i_k}(x) = 0$ for all i_k. Under this assumption, we show that by upper-bounding the maximum step-size η_{max}, SGD with non-monotone line-search has an $O(\frac{1}{T})$ convergence rate where T is the number of iterations.

Theorem 3 *(Non-Convex) Suppose that the function f satisfies the SGC with constant κ, and f_i are L_i smooth functions. Then, Algorithm 1 with $c > 1 - \frac{L_{max}}{\kappa L}$ and setting $\eta_{max} < \frac{2}{\kappa L}$ achieves the rate:*

$$\min_{k=0,1,\dots,T-1} \mathbb{E}\|\nabla f(x_k)\|^2 \leq \frac{1}{\delta T}(f(x_0) - f(x^*)) \tag{29}$$

where $\delta = (\eta_{max} + \frac{2(1-c)}{L_{max}}) - \kappa(\eta_{max} - \frac{2(1-c)}{L_{max}} + L\eta_{max}^2)$.

Proof The proof is similar to the proof of Theorem 3 in [15]. We present it for completeness. Note that for two variables a and b, we have:

$$\|a-b\|^2 = \|a\|^2 + \|b\|^2 - 2\langle a, b\rangle$$
$$\Rightarrow -2\langle a, b\rangle = \frac{1}{2}(\|a-b\|^2 - (\|a\|^2 + \|b\|^2)$$

We define $\Delta_k = f(x_{k+1}) - f(x_k)$, The L-smoothness of f implies that:

$$\Delta_k \leq \langle \nabla f(x_k), x_{k+1} - x_k\rangle + \frac{L}{2}\|x_{k+1} - x_k\|^2$$

Using $x_{k+1} = x_k - \eta_k \nabla f_{i_k}(x_k)$, we conclude that:

$$\begin{aligned}
\Delta_k &\leq -\eta_k \langle \nabla f(x_k), \nabla f_{i_k}(x_k)\rangle + \frac{L}{2}\eta_k^2 \|\nabla f_{i_k}(x_k)\|^2 \\
&= \frac{\eta_k}{2}(\|\nabla f(x_k) - \nabla f_{i_k}(x_k)\|^2 - \|\nabla f(x_k)\|^2 - \|\nabla f_{i_k}(x_k)\|^2) + \frac{L}{2}\eta_k^2 \|\nabla f_{i_k}(x_k)\|^2 \\
\Rightarrow 2\Delta_k &\leq \eta_k \|\nabla f(x_k) - \nabla f_{i_k}(x_k)\|^2 - \eta_k(\|\nabla f(x_k)\|^2 + \|\nabla f_{i_k}(x_k)\|^2) + L\eta_k^2 \|\nabla f_{i_k}(x_k)\|^2
\end{aligned}$$

We put $L_{max} = \max L_i$, it implies that

$$\eta_{min} = \min\{\frac{2(1-c)}{L_{i_k}}, \eta_{max}\} \leq \min\{\frac{2(1-c)}{L_{max}}, \eta_{max}\} \leq \eta_k \leq \eta_{max}$$

It leads that:

$$\begin{aligned}
2\Delta_k \leq\; &\eta_{max}\|\nabla f(x_k) - \nabla f_{i_k}(x_k)\|^2 - \eta_{min}(\|\nabla f(x_k)\|^2 + \|\nabla f_{i_k}(x_k)\|^2) \\
&+ L\eta_{max}^2 \|\nabla f_{i_k}(x_k)\|^2
\end{aligned}$$

Taking expectations with respect to $\nabla f_{i_k}(x_k)$, we have:

$$2\mathbb{E}[\Delta_k] \leq \eta_{max}\mathbb{E}[\|\nabla f(x_k) - \nabla f_{i_k}(x_k)\|^2] - \eta_{min}\mathbb{E}\left[(\|\nabla f(x_k)\|^2 + \|\nabla f_{i_k}(x_k)\|^2)\right]$$

$$\begin{aligned}
&+\ L\eta_{max}\mathbb{E}[\|\nabla f_{i_k}(x_k)\|^2] \\
&\leq\ \eta_{max}\mathbb{E}[\|\nabla f_{i_k}(x_k)\|^2] - \eta_{max}\|\nabla f(x_k)\|^2 - \eta_{min}\mathbb{E}\left[(\|\nabla f(x_k)\|^2 + \|\nabla f_{i_k}(x_k)\|^2)\right] \\
&+\ L\eta_{max}\mathbb{E}[\|\nabla f_{i_k}(x_k)\|^2] \\
&\leq\ [\eta_{max}(\kappa-1) - \eta_{min}(1+\kappa) + L\eta_{max}^2\kappa]\|\nabla f(x_k)\|^2 \\
&\leq\ (-(\eta_{max}+\eta_{min}) - \kappa(\eta_{max} - \eta_{min} + L\eta_{max}^2))\|\nabla f(x_k)\|^2
\end{aligned}$$

Now, we set $\delta = (\eta_{max}+\eta_{min}) - \kappa(\eta_{max} - \eta_{min} + L\eta_{max}^2)$. It implies that:

$$\|\nabla f(x_k)\|^2 \leq \frac{2}{\delta}\mathrm{E}[-\Delta_k]$$

Taking expectations and summing from $k=0$ to K

$$\begin{aligned}
\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(x_k)\|^2] &\leq \frac{2}{\delta K}\sum_{k=0}^{K-1}\mathbb{E}[-\Delta_k] \\
&= \frac{2}{\delta K}\sum_{k=0}^{K-1}\mathbb{E}[f(x_k) - f(x_{k+1})] \\
&= \frac{2}{\delta K}\mathbb{E}[f(x_0) - f(x_{k+1})] \\
&\leq \frac{2}{\delta K}\mathbb{E}[f(x_k) - f(x^*)]
\end{aligned}$$

So, we have:

$$\min\mathbb{E}[\|\nabla f(x_k)\|^2] \leq \frac{2}{\delta T}[f(x_k) - f(x^*)]$$

Now, we show that δ is positive. To this end, we consider the following cases:

- If $\eta_{max} \leq \frac{2(1-c)}{L_{max}}$. It implies that $\eta_{max} = \eta_{min}$, therefore, we have:

$$\begin{aligned}
\delta &= (\eta_{max}+\eta_{min}) - \kappa(\eta_{max} - \eta_{min} + L\eta_{max}^2) \\
&= 2\eta_{max} - \kappa L\eta_{max}^2 \\
\Rightarrow \eta_{max} &< \frac{2}{L\kappa}
\end{aligned}$$

- If $\eta_{max} > \frac{2(1-c)}{L_{max}}$. It implies that $\eta_{min} = \frac{2(1-c)}{L}$. Therefore, we have:

$$\delta = (\eta_{max} + \frac{2(1-c)}{L}) - \kappa(\eta_{max} - \frac{2(1-c)}{L} + L\eta_{max}^2) \tag{30}$$

Now, we can find an interval such that κ is positive such as:

$$\eta_{max} \in \left(0, \frac{(1-\kappa) + \sqrt{(\kappa-1)^2 + \frac{L(1-c)(1+\kappa)8\kappa}{L_{max}}}}{2L\kappa}\right) \tag{31}$$

On the other hand, we know that $\eta_{max} > \frac{2(1-c)}{L_{max}}$. So, we have:

$$\begin{aligned}
\frac{(1-\kappa)+\sqrt{(\kappa-1)^2+\frac{L(1-c)(1+\kappa)8\kappa}{L_{max}}}}{2L\kappa} &> \frac{2(1-c)}{L_{max}} \\
\frac{L(1-c)(1+\kappa)8\kappa}{L_{max}} &> \left(\frac{4\kappa L(1-c)}{L_{max}}+(1-\kappa)\right)^2-(\kappa-1)^2 \\
\frac{L_{max}}{\kappa L} &> (1-c) \\
c &> 1-\frac{L_{max}}{\kappa L}
\end{aligned}$$

Therefore, we have: $c \in (1-\frac{L_{max}}{\kappa L}, 1)$.
Substituting the maximum value for c into the upper-bound on η_{max} yields a similar requirement,

$$\eta_{max} \in (0, \frac{2}{\kappa L})$$

Putting the two cases together gives the final constraints on c and η_{max} as

$$c \geq 1-\frac{L_{max}}{\kappa L}, \qquad \eta_{max} < \frac{2}{\kappa L} \tag{32}$$

We note that the upper and lower bounds on η_k are consistent since:

$$\eta_{min} = \min\left\{\frac{2(1-c)}{L_{max}}, \eta_{max}\right\} < \min\left\{\frac{2L_{max}}{\kappa L L_{max}}, \eta_{max}\right\} = \max\left\{\frac{2}{\kappa L}, \eta_{max}\right\} \leq \frac{2}{\kappa L}$$

□

5 Experimental Evaluation

We solve the logistic regression problem, a standard test problem in the machine learning area. This problem is to minimize $F(x)$ given by:

$$F(x) = \frac{1}{n}\sum_{i=1}^{n}\log(1+e^{-y_i<A_i,x>}) \tag{33}$$

where n is the number of samples, $y_i \in \{-1, 1\}$, A_i is the input feature vector for the i-th data-point and $< A_i, x >$ refers to the dot product. Table 1 summarizes the datasets used [1].

The following initial values are used for the parameters, $N = 5, \rho = 0.25, \beta_2^{RMSProp} = 0.35, \beta_{Mom} = 0.75, \eta_0 = 1, \omega_b = 1,$ mini-batch size $= 100$. We use the following values for the non-monotone parameter for Algorithm 1, $\gamma_k \in \{0.05, 0.15, 0.25, 0.75, 1-0.75\exp(-\|\nabla f_{i_k}(x_k)\|)\}$.

Figures 2, 3, and 4 show the loss function for the Adagrad [5], RMSProp [6] and SGD [2] algorithms. The proposed method (NMSGD) has better performance in most cases.

[1] ”https://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/binary.html”

Name	number of samples	number of features
W8a	49749	300
Ijcnn1	49990	22
Segment	2310	25
Phishing	11055	68
Mushrooms	8124	112
Pendigits	10992	17
Svmguide1	3088	5

Table 1 Data Sets

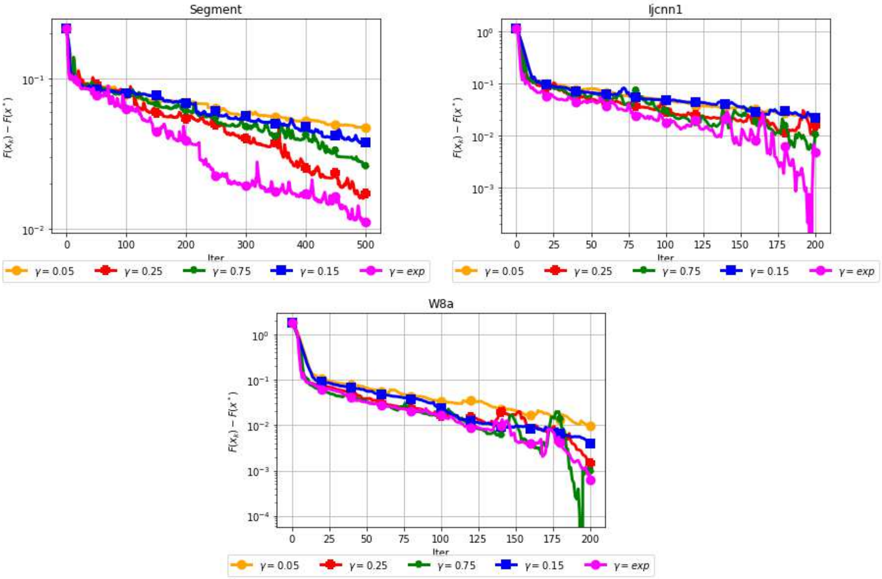

Fig. 2 γ_k v/s Loss for Adagrad direction

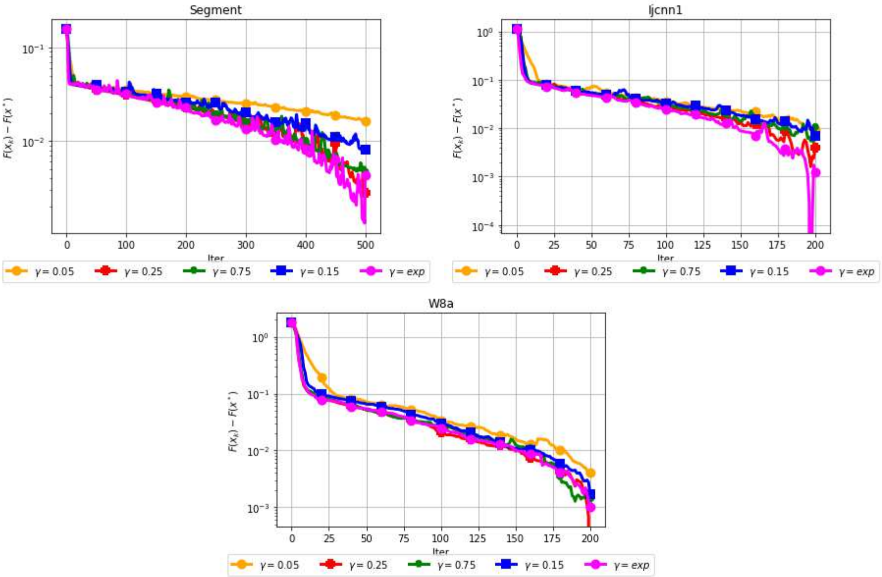

Fig. 3 γ_k v/s Loss for RMSProp direction

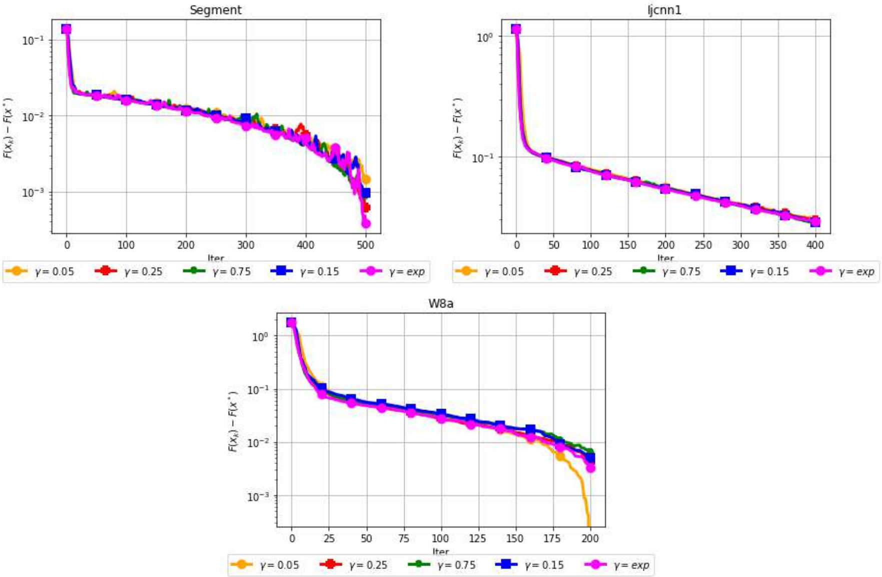

Fig. 4 γ_k v/s Loss for SGD direction

The norm of the gradient for the SGD algorithm is in Fig 5.

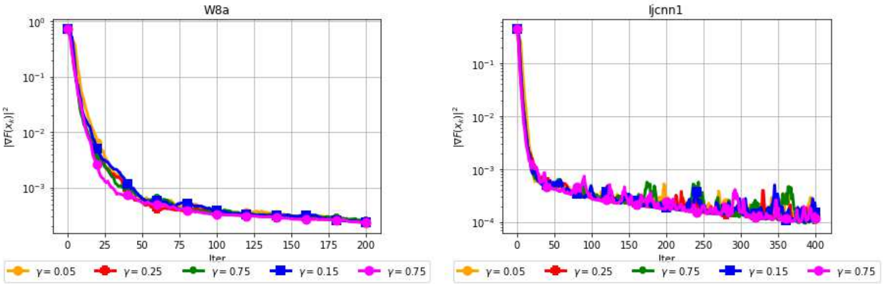

Fig. 5 The square norm of gradient - γ_k v/s SGD search direction.

Search direction: We used four different search directions, SGD [2], Adagrad [5], RMSProp [6], and momentum search direction in our implementation of NMSGD. We use only the non-monotone line search step size for these search directions. The resulting loss function and norm of the gradient are shown in Fig 6 and 7. As we can see, the Adagrad [5] and RMSProp [6] provide the best search direction for our method NMSGD.

Step size: To demonstrate the algorithm's (NMSGD) efficiency compared to other available step sizes, we implement the SGD [2] and RMSProp [6] algorithms with different step sizes. We use the step size generated by the non-monotone line search method, Armijo line search method ("line") [15], Polyak step size ("SPS") [12], BB method ("BB") [13], and constant step size ($\eta_k = 0.05$). The loss functions for SGD and RMSProp algorithm are in Figures 8 and 9.

The values of the norm of the gradient are in Fig. 10.

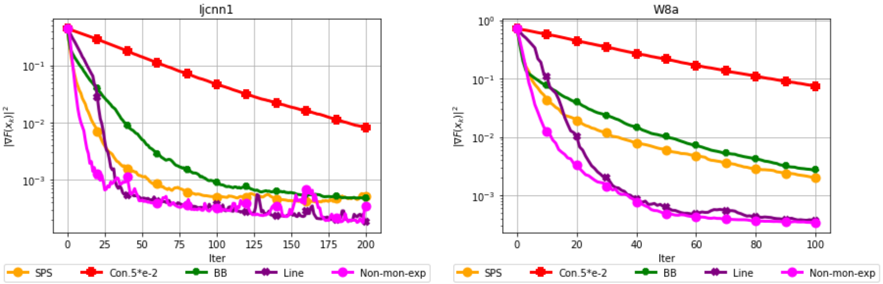

Fig. 10 Norm of the gradient for different learning rate for SGD.

6 Discussion

The line search method for SGD proposed by [15] may not evaluate an effective value for the step size when the norm of gradient is too large. In this case, the step size generated by Armijo line search algorithm [15] is extremely tiny and the algorithm needs a huge number of iterations. To deal with the issue, this

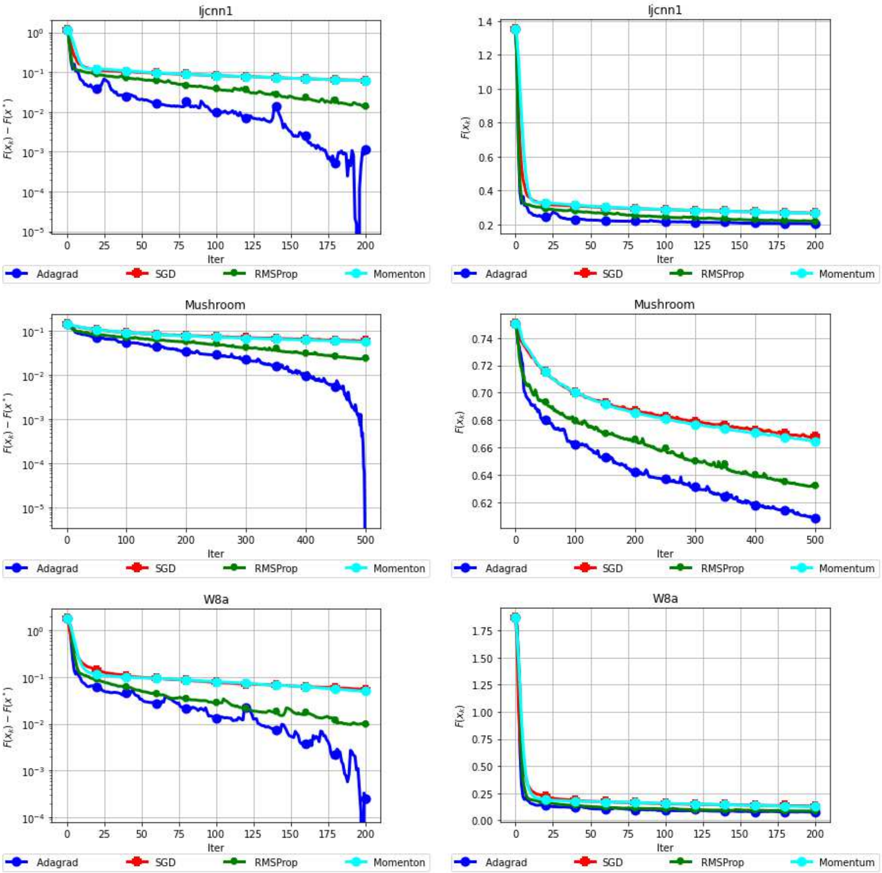

Fig. 6 Loss function for different search directions

paper introduces the non-monotone line search method for SGD that improves the proposed algorithm in [15]. In the non-monotone strategy some growth in the function value is permitted, which leads the non-monotone line search to choose a larger step size at each iteration than the Armijo line search method. This dramatically decreases the number of iterations of the algorithm than the Armijo line search proposed in [15]. The existing schemes for determining the parameter γ_k work based on the number of iterations [19] or constant values, and may not reduce the value of the objective function significantly in initial iterations. To overcome this drawback, we propose a new scheme for choosing γ_k based on the gradient behaviour of the objective function. This can reduce the total number of iteration.

Although the numerical results presented in this paper were done for regression problems, different cases have been investigated by us. In those additional experiments (not reported here), it is the case that Eq (13) leads to better results than other values for parameter γ_k, and it can also be concluded that for

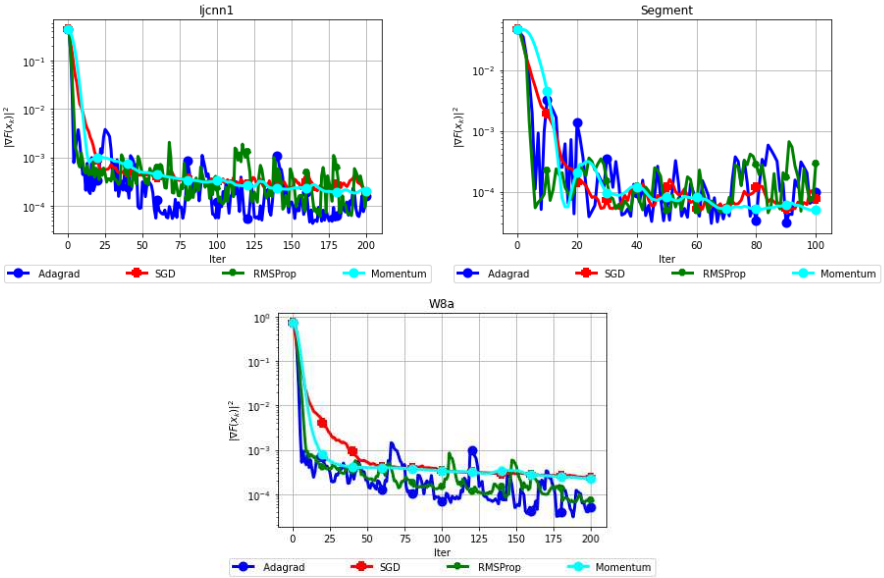

Fig. 7 Norm of the gradient for different search directions

values close to zero (Armijo method), the algorithm has a slower convergence rate than other values of γ_k. We then examined the behavior of the non-montone method with four other algorithms, i.e., SGD, Adagrad, RMSProp, and Momentum. Finally, we showed the efficiency of the proposed method compared to other existing step sizes.

We now provide a bit more context on the design of our experiments and what inferences we can draw from the data in the manuscript. Our design was to conduct experiments which demonstrates that the proposed method is more efficient (compared to state of the art) for parameter γ_k in (13) using different search directions and demonstrate that the proposed method improves on the basic Armijo line search. Additionally, we investigate the method's performance based on the method used to determine the search direction and its efficiency compared to other methods for computing step sizes using two different search directions.

We conclude the following from Figures 2–10. Figure 2 is the behaviour of the AdaGrad plus non-monotone line search as a function of γ_k. When this parameter is 0, the AdaGrad method with non-monotone line search reverts to Armijo line search. Thus, Fig. 2 demonstrates that the non-monotone strategy improves Armijo's algorithm, and an adaptive strategy for computing γ_k improves the convergence of the method. Figure 3 is for another non-monotone search added to RMSProp for different values of γ_k. This figure demonstrates again that the non-monotone strategy is better than the Armijo line search. Figure 4 is for stochastic gradient descent with a non-monotone line search. For most datasets, all values of γ_k work well. Only for two data sets very low

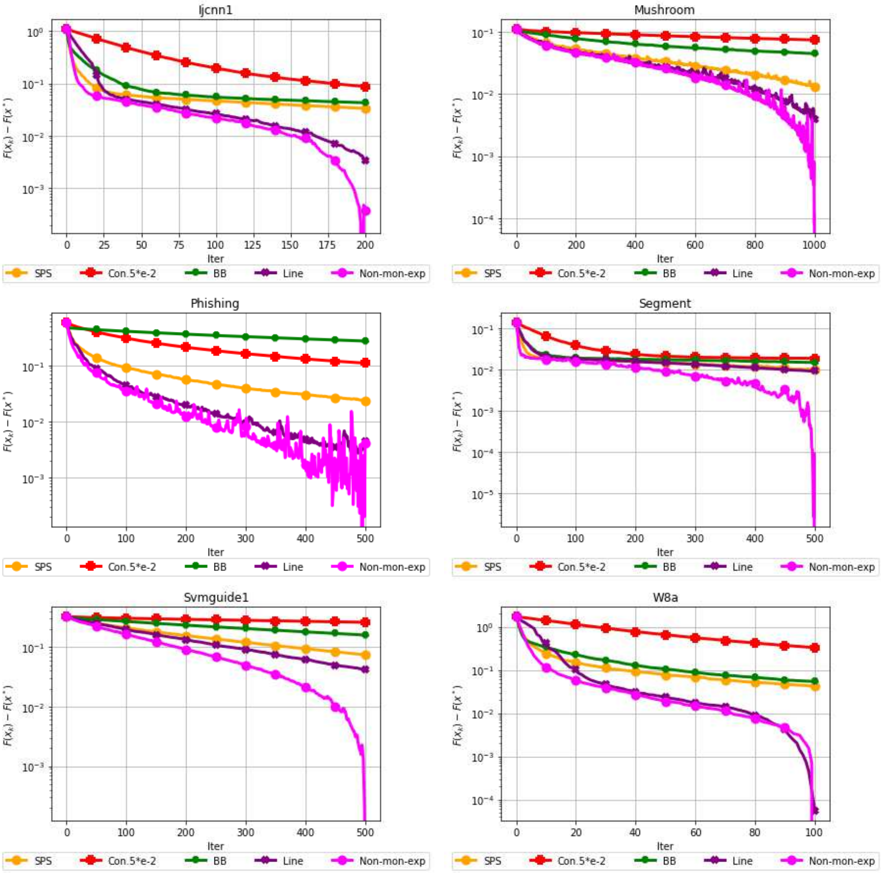

Fig. 8 Loss function v/s Learning rate for SGD.

value of $\gamma = 0.05$ does not work well. Figure 6 tells us which method (of Adagrad, SGD, RMSprop, Momentum) benefits from the non-monotone strategy. AdaGrad and RMSprop perform better compared to the other two techniques. The norm of the gradient is a better search direction to use. Next, we study the issue of determination of the step size in the method proposed here. Four popular methods for determining the step sizes are used in the comparison. These four variants of the method are compared to four variants of the SGD and RMSProp (based on Figures 8 and 9). Again, these experiments provide evidence that the non-monotone strategy with the right step size and directions performs much better than the Armijo line search.

7 Conclusion

We described a non-monotone linear search for SGD and gave an efficient and adaptive method for computing the learning rate. This method calculates the learning rate based on the gradient. We proved that the proposed algorithm

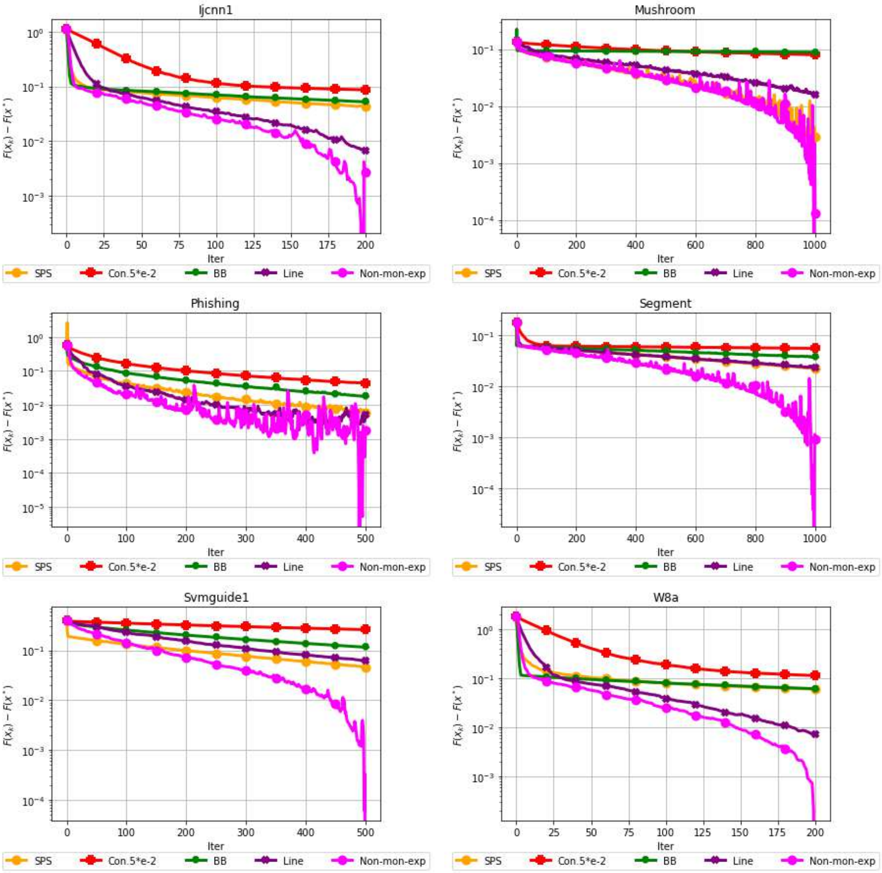

Fig. 9 Loss function v/s Learning rate for RMSProp.

has a convergence guarantee for strong convex, convex, and non-convex functions. We implemented and tested the proposed algorithm with other existing schemes on standard data sets on linear regression problems in machine learning. The experimental results show that the proposed algorithm improves the Armijo line search method in [15] in most cases that were tested.

Declarations

Acknowledgements

The authors would like to thank the editors and anonymous reviewers for their constructive comments.

Funding

No funding was received for conducting this study.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Availability of data and material

The dataset used in this paper is available from the following link https://www.openml.org/search?q=gisette&type=data&sort=runs&status=active

Conflict of Interests

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Competing interests

The authors have declared that no competing interests exist.

Authors' contributions

All authors contributed equally to this work.

References

[1] Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, ??? (1999)

[2] Robbins, H., Monro, S.: A stochastic approximation method. The annals of mathematical statistics, 400–407 (1951)

[3] Nemirovski, A., Juditsky, A., Lan, G., Shapiro, A.: Robust stochastic approximation approach to stochastic programming. SIAM Journal on optimization **19**(4), 1574–1609 (2009)

[4] Schaul, T., Zhang, S., LeCun, Y.: No more pesky learning rates. In: Dasgupta, S., McAllester, D. (eds.) Proceedings of the 30th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 28, pp. 343–351. PMLR, Atlanta, Georgia, USA (2013)

[5] Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of machine learning research **12**(7) (2011)

[6] Tieleman, T., Hinton, G.: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. Technical Report (2017)

[7] Moulines, E., Bach, F.: Non-asymptotic analysis of stochastic approximation algorithms for machine learning. Advances in neural information processing systems **24** (2011)

[8] Needell, D., Ward, R., Srebro, N.: Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. Advances in neural information processing systems **27** (2014)

[9] Gower, R.M., Loizou, N., Qian, X., Sailanbayev, A., Shulgin, E., Richtárik, P.: Sgd: General analysis and improved rates. In: International Conference on Machine Learning, pp. 5200–5209 (2019). PMLR

[10] Ghadimi, S., Lan, G.: Stochastic first-and zeroth-order methods for nonconvex stochastic programming. SIAM Journal on Optimization **23**(4), 2341–2368 (2013)

[11] Karimi, H., Nutini, J., Schmidt, M.: Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 795–811 (2016). Springer

[12] Loizou, N., Vaswani, S., Laradji, I.H., Lacoste-Julien, S.: Stochastic polyak step-size for sgd: An adaptive learning rate for fast convergence. In: International Conference on Artificial Intelligence and Statistics, pp. 1306–1314 (2021). PMLR

[13] Tan, C., Ma, S., Dai, Y.-H., Qian, Y.: Barzilai-borwein step size for stochastic gradient descent. Advances in neural information processing systems **29** (2016)

[14] Yang, Z.: On the step size selection in variance-reduced algorithm for nonconvex optimization. Expert Systems with Applications **169**, 114336 (2021)

[15] Vaswani, S., Mishkin, A., Laradji, I., Schmidt, M., Gidel, G., Lacoste-Julien, S.: Painless stochastic gradient: Interpolation, line-search, and convergence rates. Advances in neural information processing systems **32** (2019)

[16] Vaswani, S., Dubois-Taine, B., Babanezhad, R.: Towards noise-adaptive, problem-adaptive (accelerated) stochastic gradient descent. ICML 2022, arXiv preprint arXiv:2110.11442 (2021)

[17] Grippo, L., Lampariello, F., Lucidi, S.: A nonmonotone line search technique for newton's method. SIAM journal on Numerical Analysis **23**(4), 707–716 (1986)

[18] Chamberlain, R., Powell, M., Lemarechal, C., Pedersen, H.: The watchdog technique for forcing convergence in algorithms for constrained optimization. In: Algorithms for Constrained Minimization of Smooth Nonlinear Functions, pp. 1–17. Springer, ??? (1982)

[19] Ahookhosh, M., Amini, K., Peyghami, M.R.: A nonmonotone trust-region line search method for large-scale unconstrained optimization. Applied Mathematical Modelling **36**(1), 478–487 (2012)

[20] Amini, K., Ahookhosh, M., Nosratipour, H.: An inexact line search approach using modified nonmonotone strategy for unconstrained optimization. Numerical Algorithms **66**(1), 49–78 (2014)

[21] Thomas, G.B., Weir, M.D., Hass, J., Heil, C., Behn, A.: Thomas' Calculus: Early Transcendentals. Pearson Boston, ??? (2010)

[22] Polyak, B.T.: Gradient methods for minimizing functionals. Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki **3**(4), 643–653 (1963)