Journal of Computational and Applied Mathematics 477 (2026) 117120

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

A line-search based SGD algorithm with Adaptive Importance Sampling☆

Filippo Camellini [a], Serena Crisci [b], Anna De Magistris [b,*], Giorgia Franchini [a]

[a] Department of Physics, Informatics and Mathematics, University of Modena and Reggio Emilia, Via Campi 213/B, Modena, 41125, Italy
[b] Department of Mathematics and Physics, University of Campania "Luigi Vanvitelli", Viale Lincoln, 5, Caserta, 81100, Italy

ARTICLE INFO

Keywords:
Machine learning
Nonlinear optimization
Stochastic gradient
Importance sampling
Line-search

ABSTRACT

Stochastic Gradient methods are widely used in the field of supervised learning associated with big data. In this context, importance sampling-based algorithms have been proposed to minimize the variance of the stochastic gradient by introducing practical strategies to approximate the optimal sampling distribution, which is otherwise only theoretically accessible.

In this paper, we propose a scheme that combines stochastic gradient descent with adaptive importance sampling with automatic step-size selection based on a stochastic Armijo-type line-search. This approach makes the method robust to the choice of the initial step-size, which would otherwise require a tuning phase that is computationally expensive or even impractical in certain big data scenarios. Moreover, we introduce different mini-batch variants to foster the practical acceleration of the original scheme. Finally, numerical experiments are presented on real datasets to validate the proposed method in the context of supervised classification problems.

1. Introduction

In this work, the following optimization problem is considered:

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

where each $f_i : \mathbb{R}^d \to \mathbb{R}$ is a continuously differentiable function and $d, n \in \mathbb{N}$. Specifically, in a supervised learning context, n represents the total number of examples in the training set, and f_i denotes the loss value associated with the ith example. Within this framework, Stochastic Gradient methods find widespread application in solving (1). Indeed, these algorithms are particularly effective in scenarios where n is very large, as often encountered in supervised learning tasks [1].

One of the most commonly employed schemes is Stochastic Gradient Descent (SGD) [2–5], where, at each iteration k, the search direction is computed by approximating the gradient of F with the gradient of a randomly chosen f_{i_k}, with i_k sampled from the discrete uniform distribution over $\{1, \dots, n\}$. This computed descent direction is known as the stochastic gradient. Specifically, the following SGD scheme (2) is applied to update the model parameters $w \in \mathbb{R}^d$ at the kth iteration:

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

☆ This article is part of a Special issue entitled: 'NAMAS-24' published in Journal of Computational and Applied Mathematics.

* Corresponding author.
E-mail addresses: filippo.camellini@unimore.it (F. Camellini), serena.crisci@unicampania.it (S. Crisci), anna.demagistris@unicampania.it (A. De Magistris), giorgia.franchini@unimore.it (G. Franchini).

https://doi.org/10.1016/j.cam.2025.117120
Received 11 March 2025; Received in revised form 9 September 2025
Available online 13 October 2025

where $\nabla f_{i_k}(w_k)$ is the stochastic gradient that approximates $\nabla F(w_k)$, and $\eta_k > 0$ is the step-size.

SGD is highly efficient and scalable, making it a popular choice in big data scenarios. However, unlike classical gradient descent methods, it has some challenges related to its asymptotic behavior. For instance, even under the assumption of strong convexity, when using a constant step-size, SGD converges only to a neighborhood of the optimal value [2].

This behavior is due to the variance associated with the stochastic gradient: reducing the variance thus leads to better convergence. Several methods inspired by this principle have been proposed in the literature. Among these, we can mention SVRG [6] and SAGA [7], hybrid methods that make use of the full gradient ∇F computation, updated only partially during iterations through ∇f_{i_k} sampled randomly. On the other hand, again in the same spirit, but sometimes with even less guarantee of convergence, methods such as AdaGrad [8], RMSProp [9] or AdaM [10] have been introduced in the literature. Another strategy consists of progressively increasing the mini-batch size used to compute the stochastic gradient. This approach, followed by the so-called dynamic sampling methods, has been employed by several algorithm in literature [2,11–16].

Other methods, on the other hand, adopt an importance sampling logic [17–19], choosing an optimal distribution that minimizes the variance associated with the stochastic gradient to sample i_k from $\{1, \dots, n\}$. To achieve this, many works propose methods based on the knowledge of the Lipschitz constant L of the gradient of the target function or an approximation of it [18]. However, from a practical perspective, having a robust estimation method for L is very challenging. This fact is demonstrated by several works in the field of optimization that employ approximation strategies for the Lipschitz constant [20,21].

To address this issue, the work in [19] proposes the use of a sampling distribution that does not rely on L, and approximates the optimal sampling distribution that theoretically minimizes the variance associated with the stochastic gradient. That work presents several experiments showing that the proposed algorithm, called SGD-AIS, achieves better performance compared to the SGD algorithm with uniform distribution.

The SGD-AIS algorithm presents two aspects in addition to the choice of an appropriate sampling distribution. The first is the use of a decreasing rule for computing the step-size η_k, given by the formula:

$$\eta_k = \frac{\eta_0}{1 + \mu k}, \tag{3}$$

where η_0 is the initial step-size and μ is a positive constant. The use of such a rule ensures convergence under the assumption of strong convexity, as shown in [19]. The second element concerns the possibility of extending the computation of the stochastic gradient to the case where multiple indices are sampled at each iteration. In this way, a mini-batch version of the stochastic gradient can be computed. This approach relies on a partitioning of the dataset indices performed before the optimization process, which remains fixed until the final iteration.

The goal and original contribution of this paper is to propose an alternative approach for the last two aspects mentioned for the SGD-AIS algorithm.

First contribution. Firstly, a method for automatic step-size selection is proposed. Automatic hyperparameter selection strategies have already been explored in the stochastic optimization literature, particularly for step-size tuning, through a variety of techniques [12,14,15,22–26]. Moreover, these strategies have also been extended to the automatic selection of other hyperparameters, such as the momentum parameter [27,28]. In this work, we draw inspiration for step-size selection from a stochastic variant of the Armijo-type line-search procedure [29], as employed by the LISA algorithm [12,13], instead of relying on the classical decreasing step-size rule (3) used in [19]. Although we adopt a similar line-search strategy, we emphasize that the proposed algorithms differ from LISA algorithm in the variance reduction technique used: rather than dynamic sampling, we rely on adaptive importance sampling proposed in [19] for the stochastic gradient computation. There are several reasons to adopt an automatic step-size selection. First of all, the selection of the initial step-size is delicate and highly dependent on the objective function and dataset [30]. Therefore, an experimental selection of this hyperparameter may be computationally expensive in certain contexts. Furthermore, the choice of the functional form of the decreasing rule represents another hyperparameter that must be established. Finally, another motivation arises from the benefits of combining the stochastic line-search condition with the importance sampling approach. Intuitively, using the proposed distribution for SGD-AIS, the most representative functions f_i among all those that make up F are sampled. Verifying the sufficient decrease condition on these sampled f_i can therefore be particularly effective.

Second contribution. The second proposal consists of an alternative version of the mini-batch SGD-AIS proposed in [19], which removes the restrictive partitioning of the dataset. In fact, the more rigid approach presented in [19] exhibits two main drawbacks: firstly, it requires additional computational cost and secondly, it does not allow for a possible future combination of importance sampling with an increasing mini-batch size strategy.

The paper is structured as follows. Section 2 describes the SGD-AIS algorithm introduced in [19]. In Section 3, we introduce the proposed variants of the SGD-AIS algorithm. Specifically, the automatic step-size selection strategy and the alternative methodology for computing the mini-batch SGD-AIS are presented. Subsequently, in Section 4, we extend the convergence results presented in [19] to both the strongly convex and non-convex cases, adapting it to the line-search version. Section 5 shows the experimental results for the binary classification test problem under consideration. In particular, $\mathbf{l}_2$-regularized versions of both convex and non-convex loss functions are considered with several datasets. Finally, Section 6 reports the conclusions.

2. Adaptive Importance Sampling algorithms

In this section, we present the algorithms based on Adaptive Importance Sampling illustrated in [19]. Specifically, Section 2.1 describes the SGD-AIS algorithm, while Section 2.2 reports its extension to the mini-batch stochastic gradient case.

2.1. SGD-AIS

Similarly to the classical SGD algorithm, the update step for methods based on importance sampling follows formula (2), where the step-size η_k is determined by the decreasing law (3). The difference lies in the distribution used to sample an index from the set $\{1, \dots, n\}$, which differs from the uniform distribution. Specifically, a probability distribution $p^k = (p_1^k, \dots, p_n^k)$ is considered at the kth iteration, where p_i^k stands for the probability to sample the ith component function. Then, in order to have an unbiased estimator of $\nabla F(w_k)$, the stochastic gradient g_k can be computed as:

$$g_k = \frac{1}{n p_{i_k}^k} \nabla f_{i_k}(w_k), \tag{4}$$

where $\nabla f_{i_k}(w_k)$ is the gradient corresponding to the sampled index i_k. In the case that p^k is the uniform distribution, (4) corresponds to the classical SGD stochastic gradient, otherwise an importance sampling approach is exploited.

As demonstrated in [31], the optimal sampling distribution $(p^k)^\star$ that minimizes the variance of g_k is given by:

$$(p_i^k)^\star = \frac{\|\nabla f_i(w_k)\|}{\sum_{j=1}^n \|\nabla f_j(w_k)\|}.$$

This reported formula is impractical because it requires the full gradient computation, which is not feasible in a big data scenario. For this reason, an adaptive importance sampling distribution is introduced in [19]. In this approach, the sampling distribution at the iteration k is computed as:

$$p_i^k = \alpha_k \frac{\pi_i^k}{\sum_{j=1}^n \pi_j^k} + (1 - \alpha_k)\frac{1}{n}, \tag{5}$$

where:

- $\alpha_k \in [\underline{\alpha}, \overline{\alpha}] \subseteq (0, 1)$ is an increasing sequence defined by the following formula:

  $$\alpha_k = \underline{\alpha} + \frac{k}{maxit}(\overline{\alpha} - \underline{\alpha}),$$

  where $maxit$ is the maximum number of iterations.
- $\pi^0 = (1, \dots, 1)$ and, at each iteration $k \geq 1$, π^k is updated only at the sampled component i_k in accordance with the following:

  $$\pi_i^k = \begin{cases} \|\nabla f_{i_k}(w_k)\| & \text{if } i = i_k, \\ \pi_i^{k-1} & \text{otherwise.} \end{cases} \tag{6}$$

The probability distribution p^k thus results as a weighted average between a uniform distribution and a reweighted sampling distribution. In particular, since the sequence α_k is increasing, the indices in the initial iterations are sampled in a way that closely resembles a uniform distribution, promoting greater exploration of the indices set. As k increases, the distribution p^k gradually shifts toward the approximation of $p^\star$.

2.2. Mini-batch SGD-AIS

An extension of SGD-AIS to the mini-batch case is described in [19]. In mini-batch SGD-AIS, several component functions are selected to compute the stochastic gradient g_k.

If m is the mini-batch size, without loss of generality assume that n is divisible by m. In case the two numbers are not each other's divisor some elements can be discarded or replicated to eliminate or, respectively, complete the last mini-batch. Then, the index set is divided into m separate subsets $\{G_1, \dots, G_m\}$, each containing $\frac{n}{m}$ items. At each iteration k, the following steps are performed to construct the mini-batch and compute the stochastic gradient g_k:

1. Let $\mathcal{N}_k = \emptyset$ be the set containing the indices sampled at iteration k.
2. For $j = 1, \dots, m$:

   - The subset G_j is considered.
   - One index $i_{k,j} \in G_j$ is randomly chosen with the sampling distribution $p^{k,j}$ which approximates the optimal distribution $(p^{k,j})^*$ restricted to G_j and given by

     $$(p_i^{k,j})^* = \frac{\|\nabla f_i(w_k)\|}{\sum_{i \in G_j} \|\nabla f_i(w_k)\|}, \quad \forall i \in G_j.$$

     In particular, $p^{k,j}$ is calculated analogously to the formula provided in (5) by applying it to the set of indices G_j only instead of to the whole set $\{1, \dots, n\}$:

     $$p_i^{k,j} = \alpha_k \frac{\pi_i^k}{\sum_{t \in G_j} \pi_t^k} + (1 - \alpha_k)\frac{1}{|G_j|}, \quad \forall i \in G_j.$$

- The sampled index $i_{k,j}$ is added to the set $\mathcal{N}_k$.

3. Once the last index has been sampled, the stochastic gradient g_k for this mini-batch version is then defined as:

$$g_k = \frac{1}{n \sum_{i \in \mathcal{N}_k} p_i^k} \sum_{i \in \mathcal{N}_k} \nabla f_i(w_k). \tag{7}$$

From the expression (7), we can make an important remark: g_k is an unbiased estimator of the full gradient $\nabla F(w_k)$, since it represents a weighted average of unbiased estimators of the form $\frac{1}{np_i^k}\nabla f_i(w_k)$. Specifically, the weights associated with each term, denoted by δ_i, are given by:

$$\delta_i = \frac{p_i^k}{\sum_{i \in \mathcal{N}_k} p_i^k}.$$

In other words, each individual gradient is weighted, ensuring that the overall estimate remains unbiased with respect to the full gradient.

3. Proposed SGD-AIS variants

We have developed two variations of SGD-AIS methods and its mini-batch version to address some limitations of these approaches.

The first variant aims to provide an approach that automatically selects the step-size. As a result, the costly process of choosing a good initial step-size is avoided. In Section 3.1, we then introduce an adaptive step-size computation using a stochastic version of the classical Armijo line-search [29].

The second proposal presented in Section 3.2 consists of a variation of the mini-batch selection methodology described in Section 2.2. Specifically, the goal is to overcome the rigid partitioning previously proposed, making the mini-batch construction process more flexible. This modification improves the computational efficiency of the algorithm. Moreover, it allows for potential extensions based on dynamic sampling approaches.

3.1. Automatic step-size selection with Armijo line-search

We propose to choose the step-size η_k by means of a stochastic version of Armijo line-search instead of the decreasing law (3). Let be $F_{\mathcal{N}_k}$ defined by the following formula:

$$F_{\mathcal{N}_k}(w) = \frac{1}{m} \sum_{i \in \mathcal{N}_k} f_i(w),$$

where $\mathcal{N}_k$ is the set of sampled indices and $m = |\mathcal{N}_k|$ is the mini-batch size. We combine the scheme (2) with a stochastic Armijo-type sufficient decrease condition:

$$F_{\mathcal{N}_k}(w_k + \eta_k d_k) \leq F_{\mathcal{N}_k}(w_k) + \gamma \eta_k \nabla F_{\mathcal{N}_k}(w_k)^\top d_k, \tag{8}$$

where $d_k = -g_k$ and g_k is the stochastic gradient in its more general mini-batch version (7). This condition is obtained by adopting a standard backtracking strategy that reduces the step-size of a prefixed factor for each backtracking iteration. Specifically, let $\beta \in (0,1)$, $\eta_{\max} \geq 1$ and $\gamma \in (0, \frac{1}{\eta_{\max}})$. Then, $\eta_k = \eta_{\max}\beta^{m_k}$ where m_k is the first nonnegative integer such that

$$F_{\mathcal{N}_k}(w_k + \eta_{\max}\beta^{m_k} d_k) \leq F_{\mathcal{N}_k}(w_k) + \gamma \eta_{\max}\beta^{m_k} \nabla F_{\mathcal{N}_k}(w_k)^\top d_k. \tag{9}$$

Algorithm 1 summarizes the line-search procedure to select η_k.

Algorithm 1 Stochastic Line-search for η_k

Require: $\eta_{\max} \geq 1$, $\beta \in (0,1)$, $\gamma \in (0, \frac{1}{\eta_{\max}})$
1: Set $\eta = \eta_{\max}$
2: **while** $F_{\mathcal{N}_k}(w_k + \eta d_k) > F_{\mathcal{N}_k}(w_k) + \gamma \eta \nabla F_{\mathcal{N}_k}(w_k)^\top d_k$ **do**
3: $\quad \eta = \eta\beta$
4: **end while**
5: $\eta_k = \eta$

Algorithm 1 is therefore used to compute the step-size in the schemes presented in Sections 2.1 and 2.2, replacing the decreasing rule. As a result, the SGD-AIS and mini-batch SGD-AIS methods with line-search are derived, which are reported in Algorithms 2 and 3, respectively. The key structural difference between the two algorithms lies in the use of mini-batch in Algorithm 3, where multiple samples are selected in parallel from disjoint groups. In terms of convergence stability, the mini-batch variant benefits from variance reduction in the stochastic gradient estimate. As a result, it exhibits a smoother and more stable loss trajectory over time. In contrast, the single-sample version (Algorithm 2) is more sensitive to the variability of individual sample gradients, which can

result in more oscillatory behavior. Overall, the use of a line-search in both algorithms ensures robustness to the choice of initial step-size, but the addition of mini-batch in Algorithm 3 provides an additional layer of stability and efficiency.

Algorithm 2 SGD-AIS with line-search

Require: Number of iterations $maxit$; Number of indices n; Weights bounds $\{\underline{\alpha}, \overline{\alpha}\}$; Algorithm 1 parameters $\eta_{\max}, \beta, \gamma$.

Initialize: w_0, $\pi_i = 1$ for all $i \in \{1, \dots, n\}$.
for $k = 0, 1, \dots, maxit$ **do**
  $\alpha_k = \underline{\alpha} + \frac{k}{maxit}(\overline{\alpha} - \underline{\alpha})$
  for $z = 1, 2, \dots, n$ **do**
    $p_z^k = \alpha_k \frac{\pi_z}{\sum_{j=1}^n \pi_j} + (1 - \alpha_k)\frac{1}{n}$
  end for
  Randomly pick $i_k \in \{1, \dots, n\}$ based on distribution p^k.
  Update $\pi_{i_k} = \|\nabla f_{i_k}(w_k)\|$.
  $g_k = \frac{1}{n p_{i_k}^k} \nabla f_{i_k}(w_k)$
  Compute η_k through Algorithm 1, with $\mathcal{N}_k = \{i_k\}$.
  $w_{k+1} = w_k - \eta_k g_k$
end for

Algorithm 3 Mini-Batch SGD-AIS with line-search

Require: Number of iterations $maxit$; Number of indices n; Weights bounds $\{\underline{\alpha}, \overline{\alpha}\}$; Mini-batch size m; Algorithm 1 parameters $\eta_{\max}, \beta, \gamma$.

Initialize: w_0, $\pi_i = 1$ for all $i \in \{1, \dots, n\}$.
Partition the dataset into m disjoint subsets $\{G_1, \dots, G_m\}$ of indices.
for $k = 0, 1, \dots, maxit$ **do**
  $\alpha_k = \underline{\alpha} + \frac{k}{maxit}(\overline{\alpha} - \underline{\alpha})$
  for $z = 1, 2, \dots, n$ **do**
    $p_z^k = \alpha_k \frac{\pi_z}{\sum_{j=1}^n \pi_j} + (1 - \alpha_k)\frac{1}{n}$
  end for
  $\mathcal{N}_k = \emptyset$
  for $j = 1, \dots, m$ **do**
    for each $i \in G_j$ **do**
      $p_i^{k,j} = \alpha_k \frac{\pi_i}{\sum_{i \in G_j} \pi_i} + (1 - \alpha_k)\frac{1}{|G_j|}$
    end for
    Randomly pick one index i_j from G_j based on local distribution $p^{k,j}$.
    Add i_j to $\mathcal{N}_k$.
    Update $\pi_{i_j} = \|\nabla f_{i_j}(w_k)\|$.
  end for
  $g_k = \frac{1}{n \sum_{i \in \mathcal{N}_k} p_i^k} \sum_{i \in \mathcal{N}_k} \nabla f_i(w_k)$
  Compute η_k through Algorithm 1.
  $w_{k+1} = w_k - \eta_k g_k$
end for

3.2. Alternative approach to the mini-batch SGD-AIS

We propose an alternative approach to the mini-batch SGD-AIS method previously presented. In this new approach, we first choose a batch size m, which defines the number of elements in each mini-batch. Unlike the previous method where the dataset was partitioned into fixed subsets, here we directly sample m indices at each iteration, based on the distribution p^k introduced in Eq. (5). The stochastic gradient g_k for this mini-batch version is then defined as:

$$g_k = \frac{1}{n \sum_{i \in \mathcal{N}_k} p_i^k} \sum_{i \in \mathcal{N}_k} \nabla f_i(w_k).$$

As also highlighted in Section 2.2, this expression ensures that the stochastic gradient remains an unbiased estimator of the full gradient. Finally, after each iteration, the probability distribution is updated to reflect the new information, allowing the algorithm to adapt and improve its sampling strategy in subsequent steps.

Algorithm 4 Free Mini-batch SGD-AIS with step-size option

Require: Number of iterations $maxit$; Number of indices n; Weights bounds $\{\underline{\alpha}, \overline{\alpha}\}$; Mini-batch size m; Choice of step-size $choice$ (d = diminishing, ls = line-search); Initial step-size η_0; Rescaling factor μ; Algorithm 1 parameters $\eta_{\max}, \beta, \gamma$.

Initialize: w_0, $\pi_i = 1$ for all $i \in \{1, \dots, n\}$.
for $k = 0, 1, \dots, maxit$ **do**
  $\alpha_k = \underline{\alpha} + \frac{k}{maxit}(\overline{\alpha} - \underline{\alpha})$
  for $z = 1, 2, \dots, n$ **do**
    $p_z^k = \alpha_k \frac{\pi_z}{\sum_{j=1}^n \pi_j} + (1 - \alpha_k)\frac{1}{n}$
  end for
  $\mathcal{N}_k = \emptyset$
  for $j = 1, 2, \dots, m$ **do**
    Randomly pick $i_j \in \{1, \dots, n\}$ based on distribution p^k.
    Add i_j to $\mathcal{N}_k$.
    Update $\pi_{i_j} = \|\nabla f_{i_j}(w_k)\|$.
  end for
  $g_k = \frac{1}{n \sum_{i \in \mathcal{N}_k} p_i^k} \sum_{i \in \mathcal{N}_k} \nabla f_i(w_k)$
  if $choice = d$ **then**
    $\eta_k = \frac{\eta_0}{(1 + \mu k)}$
  else
    Compute η_k through Algorithm 1.
  end if
  $w_{k+1} = w_k - \eta_k g_k$
end for

4. Convergence results

In this section, we establish convergence guarantees both under strong convexity and in the non-convex setting. In particular, following the approach presented in [19], we consider the case where the stochastic gradient g_k is computed as (4) and prove an analogous result to Theorem 2 stated in Section III.B. Specifically, by leveraging the boundedness of the step-size ensured by the line-search procedure as shown in Lemma 1, we obtain an estimate of the expected optimality gap $\mathbb{E}\left[F(w_k) - F^\star\right]$ in the case of strong convexity assumption.

Subsequently, we also analyze the case in which the objective function is non-convex. In particular, assuming the same hypotheses as in Theorem 2, except for the strong convexity assumption, we establish a convergence result for the average of the expected squared gradient norm. This result is analogous to Theorem 4 stated in Section IV of the supplementary material of [19].

The same assumptions as in [19] are considered:

Assumption 1. F is lower bounded by F^* and $w^* \in \arg\min_w F(w) \neq \emptyset$.

Assumption 2. Any function f_i is L_i-smooth, i.e. for each $w_1, w_2 \in \mathbb{R}^d$, $\exists L_i > 0$ such that:

$$\|\nabla f_i(w_1) - \nabla f_i(w_2)\| \leq L_i \|w_1 - w_2\|.$$

As a consequence, the objective function F is L-smooth with $L \leq \frac{1}{n}\sum_{i=1}^n L_i$ and $L \leq \max L_i$.

Assumption 3. Sequence $\{w_k\}_k \subseteq \Omega$, where $\Omega \subseteq \mathbb{R}^d$ is a compact set.

Assumption 4. Define G as:

$$G = \max_{w \in \Omega} \left\{ \max_{i=1,\dots,n} \|\nabla f_i(w)\| \right\}.$$

Since Ω is compact, G is well-defined, i.e. $G < +\infty$.

There exist some constants $\delta > 0$ and $\rho > 0$ such that:

$$\min_{w \in \Omega} \left\{ \frac{1}{n} \sum_{i=1}^n \|\nabla f_i(w)\| \right\} \geq \delta G$$

and

$$\min_{w \in \Omega} \left\{ \frac{1}{2n^2} \sum_{j=1}^n \sum_{i=1}^n (\|\nabla f_i(w)\| - \|\nabla f_j(w)\|)^2 \right\} \geq \rho G^2.$$

As stated in [19, Assumption 3, Section III], it can be shown that Assumption 4 holds when the function f_i is defined as

$$f_i(w) = g(w^\top x_i) + \frac{\lambda}{2}\|w\|^2,$$

where g is a continuous loss function and each example x_i follows either a Gaussian or a uniform distribution.

Assumption 5. Suppose $N \geq n$. Then, for all $j \in \{1, \dots, n\}$ and iteration $k \geq N$, j has been picked in the last N iteration through optimal sampling distribution defined in (5), i.e. $j \in \{i_{k-1}, \dots, i_{k-N}\}$.

Assumption 5 guarantees that, within any block of N consecutive iterations, all components in the vector π_i^k defined in (6) are updated at least once. In [19, Proposition B.1, Supplementary material] is shown that this assumption holds with high probability for large enough N, in particular

$$N \geq 4n \log \frac{n}{1-\bar{\alpha}}.$$

First, we prove the well-definiteness of the Armijo procedure defined in Algorithm 1, and provide a strictly positive lower bound for the step-size η_k. In particular, the proof of Lemma 1 proceeds along the same lines as the proof in [13, Lemma 2.2].

Lemma 1. *Under the Assumption 2, the line-search procedure defined in Algorithm 1 with the sampling probability p_i^k computed as in (5) is well-defined, and*

$$0 < \eta_{\min} \leq \min\left\{\eta_{\max}, 2\beta(1-\gamma)\frac{(1-\bar{\alpha})}{L_{\mathcal{N}_k}}\right\} \leq \eta_k \leq \eta_{\max}$$

where $\eta_{\min} = \min\left\{\eta_{\max}, 2\beta(1-\gamma)\frac{(1-\bar{\alpha})}{L_{\max}}\right\}$.

Proof. If $\eta_{\max}$ satisfies the Armijo condition (8), then $\eta_k = \eta_{\max}$. In view of the Descent Lemma [32, Lemma 6.9.1] applied to $F_{\mathcal{N}_k}$, we have

$$\begin{aligned} F_{\mathcal{N}_k}(w_k + \eta d_k) &\leq F_{\mathcal{N}_k}(w_k) + \eta \nabla F_{\mathcal{N}_k}(w_k)^T d_k + \frac{\eta^2 L_{\mathcal{N}_k}}{2}\|d_k\|^2 \\ &= F_{\mathcal{N}_k}(w_k) + \eta \nabla F_{\mathcal{N}_k}(w_k)^T d_k + \frac{\eta^2 L_{\mathcal{N}_k}}{2}(-g_k^T) d_k \\ &= F_{\mathcal{N}_k}(w_k) + \eta \nabla F_{\mathcal{N}_k}(w_k)^T d_k - \frac{\eta^2 L_{\mathcal{N}_k}}{2}\frac{m}{n\sum_{i\in\mathcal{N}_k} p_i^k}\nabla F_{\mathcal{N}_k}(w_k)^T d_k \\ &= F_{\mathcal{N}_k}(w_k) - \eta\left(1 - \frac{\eta L_{\mathcal{N}_k}}{2}\frac{m}{n\sum_{i\in\mathcal{N}_k} p_i^k}\right)\nabla F_{\mathcal{N}_k}(w_k)^T g_k \end{aligned} \tag{10}$$

As a consequence, the inequality at step 2 of Algorithm 1 is satisfied if

$$1 - \frac{\eta L_{\mathcal{N}_k}}{2}\frac{m}{n\sum_{i\in\mathcal{N}_k} p_i^k} \geq \gamma \iff \eta \leq 2(1-\gamma)\frac{n\sum_{i\in\mathcal{N}_k} p_i^k}{m L_{\mathcal{N}_k}}$$

Let assume that $\eta_{\max}$ does not satisfy the sufficient decrease condition prescribed by the Armijo inequality, then the backtracking condition ensures the existence of $\ell > 0$ such that

$$\eta_{\max}\beta^\ell \leq 2(1-\gamma)\frac{n\sum_{i\in\mathcal{N}_k} p_i^k}{m L_{\mathcal{N}_k}},$$

and hence $\eta_k = \eta_{\max}\beta^\ell$. This implies that $\frac{\eta_k}{\beta} = \eta_{\max}\beta^{\ell-1}$ does not satisfy the Armijo inequality, that means

$$\frac{\gamma\eta_k}{\beta} > \frac{F_{\mathcal{N}_k}(w_k) - F_{\mathcal{N}_k}(w_k - \frac{\eta_k}{\beta}g_k)}{\nabla F_{\mathcal{N}_k}(w_k)^T g_k} \geq \frac{\eta_k}{\beta}\left(1 - \frac{\eta_k L_{\mathcal{N}_k}}{2\beta}\frac{m}{n\sum_{i\in\mathcal{N}_k} p_i^k}\right)$$

where the last inequality descends from (10). This yields

$$\begin{aligned} \eta_k &> 2\beta(1-\gamma)\frac{n\sum_{i\in\mathcal{N}_k} p_i^k}{m L_{\mathcal{N}_k}} \geq 2\beta(1-\gamma)\frac{(1-\bar{\alpha})}{L_{\mathcal{N}_k}} \\ &\geq \min\left\{\eta_{\max}, 2\beta(1-\gamma)\frac{(1-\bar{\alpha})}{L_{\mathcal{N}_k}}\right\}, \end{aligned}$$

where the second inequality follows from (5), which ensures that for each $i \in \{1, \dots, n\}$, $k \in \mathbb{N}$,

$$p_i^k \geq \frac{(1-\bar{\alpha})}{n}.$$

Since $L_{\mathcal{N}_k} \leq L_{\max} := \max_i L_i$, we have

$$0 < \eta_{\min} = \min\left\{\eta_{\max}, 2\beta(1-\gamma)\frac{(1-\bar{\alpha})}{L_{\max}}\right\} \leq \eta_k \leq \eta_{\max}. \quad \square$$

Now, we recall Theorem 1 presented in [19, Section III.A], which is used to prove the new convergence result corresponding to Theorem 2 in this section. Theorem 1 shows that the method with adaptive importance sampling is a variance reduction method. Specifically, it follows from the statement that the variance associated with the stochastic gradient computed using the sampling distribution (5) is lower than that associated with the stochastic gradient computed using a uniform distribution.

Theorem 1 (*[19, Section III.A]*). *Under Assumptions 2–5, let $k \geq N$. Suppose that the step-size η_k satisfies the following:*

$$0 < \eta_k < \frac{(1-\bar{\alpha})^3 \underline{\alpha}\delta\rho}{(1-\bar{\alpha})^2 \underline{\alpha} N L\rho + \bar{\alpha} N L} := \bar{\eta}.$$

Then, we have:

$$Var_{i\sim p^k}\left[\frac{1}{np_i^k}\nabla f_i(w_k)\right] \leq Var_{i\sim \mathcal{U}}\left[\nabla f_i(w_k)\right] - \nu G^2, \tag{11}$$

where $\mathcal{U}$ is the uniform distribution over $\{1,\dots,n\}$ and $\nu \in (0,1)$ is defined as:

$$\nu = \underline{\alpha}\rho - \frac{\bar{\alpha} L \eta_k N}{(1-\bar{\alpha})^3\delta - (1-\bar{\alpha})^2 L\eta_k N}.$$

Before stating our main results, let us introduce the following definitions and well-known properties concerning L-smooth and strongly convex functions, which we report for the sake of clarity:

Notation 1. *Given a random variable $X \in \mathbb{R}^d$ which depends on index $i \sim p^k$, the conditional expected value of X with respect to the σ-algebra $\mathcal{F}_k$ generated by $w_0, w_1, \dots, w_k$ is denoted with $\mathbb{E}_{i\sim p^k}[X|\mathcal{F}_k]$. Instead, $\mathbb{E}[X]$ denotes the total expected value of the random variable X.*

Definition 1. A function $F : \mathbb{R}^d \to \mathbb{R}$ is σ-strongly convex if there exists a constant $\sigma > 0$ such that

$$F(w_1) \geq F(w_2) + \nabla F(w_2)^\top (w_1 - w_2) + \frac{\sigma}{2}\|w_1 - w_2\|^2, \quad \forall w_1, w_2 \in \mathbb{R}^d. \tag{12}$$

Property 1. *If the function F is L-smooth, it values the following:*

$$F(w_1) \leq F(w_2) + \nabla F(w_2)^\top (w_1 - w_2) + \frac{L}{2}\|w_1 - w_2\|^2, \quad \forall w_1, w_2 \in \mathbb{R}^d. \tag{13}$$

This property is also referred to as Descent Lemma [33, Lemma 6.9.1].

Property 2. *If the function F is σ-strongly convex and lower bounded by $F^\star$, for all $w \in \mathbb{R}^d$ it values the following:*

$$2\sigma(F(w) - F^\star) \leq \|\nabla F(w)\|^2. \tag{14}$$

A proof of this property can be found in [2, Appendix B].

Property 3. *If the function F is both σ-strongly convex and L-smooth, it follows directly from (3) and (14) that the constants σ and L satisfy the following equality:*

$$\sigma \leq L. \tag{15}$$

We now prove our main convergence result, which holds under the assumption of strong convexity.

Theorem 2. *Under Assumptions 1–5, suppose that F is a σ-strongly convex function. Suppose that the sequence $\{w_k\}_{k\in\mathbb{N}}$ is generated by Algorithm 2, whereby the step-size η_k is computed by the line-search procedure in Algorithm 1. Moreover, suppose that $\eta_{\max}$ is such that:*

$$\eta_{\max} < \min\left\{\bar{\eta}; \frac{1}{L}\right\} := \tilde{\eta}.$$

Then, it values the following:

$$\mathbb{E}\left[F(w_k) - F^\star\right] \leq (1-\sigma\eta_{\min})^{k-N}\left(\mathbb{E}\left[F(w_N)\right] - F^\star\right) + M \cdot \frac{1-(1-\sigma\eta_{\min})^{k-N}}{\sigma\eta_{\min}}, \tag{16}$$

where $M = \frac{\eta_{\max}}{2}(1-\nu)G^2$ and $0 < (1-\sigma\eta_{\min}) < 1$, with $\eta_{\min} > 0$ given by Lemma 1.

Proof. By bounding the difference $\left(F(w_{k+1}) - F(w_k)\right)$, we obtain that:

$$\begin{aligned}
F(w_{k+1}) - F(w_k) &\leq \nabla F(w_k)^\top (w_{k+1} - w_k) + \frac{L}{2}\|w_{k+1} - w_k\|^2 \\
&= -\eta_k \nabla F(w_k)^\top g_k + \frac{L}{2}\eta_k^2 \|g_k\|^2 \\
&= \frac{\eta_k}{2}\left(\|\nabla F(w_k) - g_k\|^2 - \|\nabla F(w_k)\|^2 - \|g_k\|^2\right) \\
&\quad + \frac{L}{2}\eta_k^2\|g_k\|^2 \\
&= \frac{\eta_k}{2}\|\nabla F(w_k) - g_k\|^2 - \frac{\eta_k}{2}\|\nabla F(w_k)\|^2 \\
&\quad - \frac{\eta_k}{2}\left(1 - L\eta_k\right)\|g_k\|^2 \\
&\leq \frac{\eta_k}{2}\|\nabla F(w_k) - g_k\|^2 - \frac{\eta_k}{2}\|\nabla F(w_k)\|^2 \\
&\leq \frac{\eta_{\max}}{2}\|\nabla F(w_k) - g_k\|^2 - \frac{\eta_{\min}}{2}\|\nabla F(w_k)\|^2,
\end{aligned} \tag{17}$$

where the first inequality follows from (13), the second follows from the definition of the algorithm, the fifth and the sixth follow from the assumption on $\eta_{\max}$ and Lemma 1.

Considering the conditional expected value with respect to the σ-algebra $\mathcal{F}_k$, by (17) it holds that:

$$\begin{aligned}
\mathbb{E}_{i\sim p^k}[F(w_{k+1}) - F^\star|\mathcal{F}_k] &\leq F(w_k) - F^\star + \frac{\eta_{\max}}{2}\mathbb{E}_{i\sim p^k}[\|\nabla F(w_k) - g_k\|^2|\mathcal{F}_k] \\
&\quad - \frac{\eta_{\min}}{2}\|\nabla F(w_k)\|^2 \\
&\leq F(w_k) - F^\star + \frac{\eta_{\max}}{2}\mathrm{Var}_{i\sim p^k}\left[g_k\right] - \frac{\eta_{\min}}{2}\|\nabla F(w_k)\|^2 \\
&\leq F(w_k) - F^\star + \frac{\eta_{\max}}{2}\left(\mathrm{Var}_{i\sim\mathcal{U}}\left[\nabla f_i(w_k)\right] - \nu G^2\right) - \frac{\eta_{\min}}{2}\|\nabla F(w_k)\|^2 \\
&= F(w_k) - F^\star + \frac{\eta_{\max}}{2}\left(\mathbb{E}_{i\sim\mathcal{U}}[\|\nabla f_i(w_k)\|^2|\mathcal{F}_k] - \|\nabla F(w_k)\|^2 - \nu G^2\right) \\
&\quad - \frac{\eta_{\min}}{2}\|\nabla F(w_k)\|^2 \\
&\leq F(w_k) - F^\star + \frac{\eta_{\max}}{2}(1-\nu)G^2 - \eta_{\min}\sigma(F(w_k) - F^\star) \\
&= (1 - \eta_{\min}\sigma)(F(w_k) - F^\star) + \frac{\eta_{\max}}{2}(1-\nu)G^2,
\end{aligned} \tag{18}$$

where the third inequality follows from Theorem 1, the fourth follows from the fact that $\mathrm{Var}_{i\sim\mathcal{U}}\left[\nabla f_i(w_k)\right] = \mathbb{E}_{i\sim\mathcal{U}}[\|\nabla f_i(w_k)\|^2|\mathcal{F}_k] - \|\nabla F(w_k)\|^2$ because $\nabla f_i(w_k)$ is an unbiased estimator of $\|\nabla F(w_k)\|^2$, the fifth follows by Assumption 4 and (14).

Taking total expectation and posing $M := \frac{\eta_{\max}}{2}(1-\nu)G^2$ it holds that:

$$\begin{aligned}
\mathbb{E}[F(w_{k+1}) - F^\star] &\leq (1 - \eta_{\min}\sigma)\mathbb{E}[F(w_k) - F^\star] + M \\
&\leq (1 - \eta_{\min}\sigma)\left[(1 - \eta_{\min}\sigma)\mathbb{E}[F(w_{k-1}) - F^\star] + M\right] + M \\
&\vdots \\
&\leq (1 - \eta_{\min}\sigma)^{k-N+1}(\mathbb{E}\left[F(w_N)\right] - F^\star) + M\sum_{j=0}^{k-N}(1 - \eta_{\min}\sigma)^j,
\end{aligned} \tag{19}$$

where inequality (17) has been applied recursively backward up to index $k = N$, due to Assumption 5.

Since $\eta_{\min}$ and σ are both positive, it follows that $1 - \eta_{\min}\sigma < 1$. Since by the Assumption on the step-size $\eta_{\min} < \frac{1}{L}$, it holds the following:

$$\eta_{\min}\sigma < \frac{\sigma}{L} \leq 1, \tag{20}$$

where the last inequality comes from (15). Therefore, from (20) it follows that $1 - \eta_{\min}\sigma > 0$.

Then, by (19), it values:

$$\begin{aligned}\mathbb{E}[F(w_{k+1}) - F^\star] &\leq (1-\eta_{\min}\sigma)^{k-N+1}(\mathbb{E}\left[F(w_N)\right] - F^\star) + M\sum_{j=0}^{k-N}(1-\eta_{\min}\sigma)^j \\ &= (1-\eta_{\min}\sigma)^{k-N+1}(\mathbb{E}\left[F(w_N)\right] - F^\star) \\ &+ M\cdot\frac{1-(1-\eta_{\min}\sigma)^{k-N+1}}{\eta_{\min}\sigma},\end{aligned}$$

so the thesis follows. □

We now comment on the result established in Theorem 2, discussing the applicability of its assumptions to the case of Algorithm 2 and analyzing the resulting convergence rate:

- *Assumption on* $\eta_{\max}$: the assumption $\eta_{\max} < \tilde{\eta}$ is generally difficult to verify in practice, as it depends on several theoretical constants. Nevertheless, similar assumptions are commonly found in many of the main convergence results for stochastic gradient methods, including those in [2,12,19,22], which, for example, rely on the Lipschitz constant L.
- *Optimality gap*: based on the estimate obtained in (16), the following inequality holds:

$$\lim_{k\to+\infty}\mathbb{E}\left[F(w_k) - F^\star\right] \leq \frac{M}{\sigma\eta_{\min}} = \eta_{\max}\cdot\frac{(1-\nu)G^2}{2\sigma\eta_{\min}} := \tilde{M}.$$

  This result does not ensure that the optimality gap converges to zero. Such a limitation is consistent with the convergence behavior of stochastic methods applied to strongly convex functions when a constant step size is employed, as discussed, for instance, in [2,19]. In particular, to guarantee convergence to zero, the step-size sequence is typically required to decay sufficiently fast to zero, for example as for the choice of $\eta_k = 1/k$. This condition cannot be enforced under a line-search-based strategy. Nevertheless, it is important to note that the asymptotic bound $\tilde{M}$ is proportional to $\eta_{\max}$. Therefore, from a theoretical perspective, choosing a smaller initial upper bound on the step size leads to a smaller limiting gap from the optimal value. On the downside, this choice may slow down the convergence in practice, resulting in a less efficient training process.

The second main result, presented in Theorem 3, concerns the non-convex case.

Theorem 3. *Under Assumptions 1–5, suppose that the sequence* $\{w_k\}_{k\in\mathbb{N}}$ *is generated by Algorithm 2, whereby the step-size* η_k *is computed by the line-search procedure in Algorithm 1. Moreover, suppose that* $\eta_{\max}$ *is such that:*

$$\eta_{\max} < \min\left\{\overline{\eta}; \frac{1}{L}\right\} := \tilde{\eta}.$$

Then, it values the following:

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(w_k)\|^2\right] \leq \frac{(K-N)\eta_{\max}}{K\eta_{\min}}G^2(1-\nu) + \frac{1}{K}(M_1 + M_2), \tag{21}$$

where

$$M_1 = \frac{2\left(\mathbb{E}\left[F(w_N)\right] - F^\star\right)}{\eta_{\min}}$$

and

$$M_2 = NG^2.$$

Proof. Considering inequality (18), we have:

$$\begin{aligned}\mathbb{E}_{i\sim p^k}[F(w_{k+1})|\mathcal{F}_k] &\leq F(w_k) + \frac{\eta_{\max}}{2}\left(\mathbb{E}_{i\sim\mathcal{U}}[\|\nabla f_i(w_k)\|^2|\mathcal{F}_k] - \|\nabla F(w_k)\|^2 - \nu G^2\right) \\ &- \frac{\eta_{\min}}{2}\|\nabla F(w_k)\|^2 \\ &\leq F(w_k) + \frac{\eta_{\max}}{2}(1-\nu)G^2 - \frac{\eta_{\min}}{2}\|\nabla F(w_k)\|^2,\end{aligned}$$

where the second inequality follows by Assumption 4.

Taking total expectation, we obtain the following:

$$\mathbb{E}[F(w_{k+1})] - \mathbb{E}[F(w_k)] \leq \frac{\eta_{\max}}{2}(1-\nu)G^2 - \frac{\eta_{\min}}{2}\mathbb{E}[\|\nabla F(w_k)\|^2],$$

Now, summing both sides from $k = N$ to $k = K-1$, in accordance with Assumption 5, and dividing by K, we obtain:

$$\begin{aligned}\frac{\mathbb{E}[F(w_K)] - \mathbb{E}[F(w_N)]}{K} &\leq \frac{K-N}{K}\frac{\eta_{\max}}{2}(1-\nu)G^2 - \frac{\eta_{\min}}{2}\frac{1}{K}\sum_{k=N}^{K-1}\mathbb{E}[\|\nabla F(w_k)\|^2] \\ &= \frac{K-N}{K}\frac{\eta_{\max}}{2}(1-\nu)G^2 + \\ &- \frac{\eta_{\min}}{2}\frac{1}{K}\left(\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla F(w_k)\|^2] - \sum_{k=0}^{N-1}\mathbb{E}[\|\nabla F(w_k)\|^2]\right),\end{aligned}$$

which leads to the inequality

$$\begin{aligned}\frac{\eta_{\min}}{2}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla F(w_k)\|^2] &\leq \frac{\mathbb{E}[F(w_N)] - \mathbb{E}[F(w_K)]}{K} + \frac{K-N}{K}\frac{\eta_{\max}}{2}(1-\nu)G^2 \\ &+ \frac{\eta_{\min}}{2}\frac{1}{K}\sum_{k=0}^{N-1}\mathbb{E}[\|\nabla F(w_k)\|^2] \\ &\leq \frac{\mathbb{E}[F(w_N)] - F^\star}{K} + \frac{K-N}{K}\frac{\eta_{\max}}{2}(1-\nu)G^2 \\ &+ \frac{\eta_{\min}}{2}\frac{1}{K}\sum_{k=0}^{N-1}\mathbb{E}[\|\nabla F(w_k)\|^2], \\ &\leq \frac{\mathbb{E}[F(w_N)] - F^\star}{K} + \frac{K-N}{K}\frac{\eta_{\max}}{2}(1-\nu)G^2 \\ &+ \frac{\eta_{\min}}{2}\frac{1}{K}NG^2,\end{aligned}$$

from which (21) follows. □

The analysis of the assumptions follows the same reasoning as in Theorem 2. Furthermore, similar observations also apply to the estimate obtained in (21). In particular, starting from (21), we consider the limit of the average of the expected squared gradient norm:

$$\lim_{K\to+\infty}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\|\nabla F(w_k)\|^2\right] \leq \eta_{\max}\cdot\frac{(1-\nu)G^2}{\eta_{\min}},$$

for which the same considerations made earlier for the optimality gap apply. In particular, it is not guaranteed that the gradient norm converges to zero, due to the use of a step-size that does not decay to zero. However, as already pointed out in the case of Theorem 2, the asymptotic bound is controlled by $\eta_{\max}$.

5. Numerical experiments

In this section, we provide experimental results to verify the effectiveness of algorithms proposed in Section 3. Specifically, the target functions and datasets used are described in Section 5.1. Section 5.2 clarifies the purpose of the numerical experiments and shows the plots and performance metrics of the methods taken into account.

5.1. Test problems and comparison measures

The tests were conducted on $\mathbf{l}_2$-regularized linear models. Specifically, consider a dataset $D = \{(x_i, y_i)\}_{i=1,\dots,n}$, where x_i represents the ith example, $y_i \in \{-1, +1\}$ denotes its corresponding label, and $w \in \mathbb{R}^d$ are the parameters of a linear classifier. The optimization problem is formulated as follows:

$$\min_{w\in\mathbb{R}^d} F(w) := \frac{1}{n}\sum_{i=1}^{n} f_i(w) + \frac{\lambda}{2}\|w\|^2, \quad \lambda > 0, \tag{22}$$

where λ is the regularization parameter that helps prevent overfitting. In this work, λ is set to 10^{-2}, following the reference methodology established in [19], which serves as the basis for our approach.

For the fidelity term f_i, four possible choices were considered:

1. *Logistic regression (LR)*:

$$f_i(w) = \log\left(1 + \exp\left(-y_i w^\top x_i\right)\right).$$

2. *Smooth hinge loss (SMH)*:

$$f_i(w) = \begin{cases} -y_i w^\top x_i & \text{if } y_i w^\top x_i \leq 0, \\ \frac{1}{2}(1 - y_i w^\top x_i)^2 & \text{if } 0 < y_i w^\top x_i < 1, \\ 0 & \text{if } y_i w^\top x_i \geq 1. \end{cases}$$

Table 1
Dataset measures.

Dataset	d	n	Test set examples
a1a	119	1445	160
cina0	132	14 430	1603
w8a	300	44 774	4975
MNIST	784	60 000	10 000

Table 2
Summary of AIS-based methods.

Method	Algorithm	Step-size
SGD-AIS opt [19]	Section 2.1	Decreasing law
SGD-AIS LS	Algorithm 2	Line-search
BATCH FIXED opt [19]	Section 2.2	Decreasing law
BATCH FIXED LS	Algorithm 3	Line-search
BATCH FREE opt	Algorithm 4, $choice = d$	Decreasing law
BATCH FREE LS	Algorithm 4, $choice = ls$	Line-search

3. *Squared hinge loss (SQH)*:

$$f_i(w) = \left(\max\{0; 1 - y_i w^\top x_i\}\right)^2.$$

4. *2-layer Neural Network (2NN)*:

$$f_i(w) = \left(\frac{1}{1 + \exp\left(-y_i w^\top x_i\right)}\right)^2.$$

With the choice of the loss functions 1, 2, 3, the target function in (22) is strongly convex, whereas with the choice of 4, it results non-convex.

For each target function, the datasets listed in Table 1 were used. The datasets a1a, w8a, and MNIST are stored in [34], while cina0 is available at [35]. The multiclass MNIST dataset was adapted to the binary case by considering as classes even and odd digits. Each dataset was isolated into training and testing to more robustly measure the performance of the training process.

In Section 5.2 two comparison metrics are considered:

- $OptGap_k$, optimality gap at iteration k which is computed as follows:

$$OptGap_k = F(w_k) - F^\star$$

  where w_k are the parameters at iteration k and $F^\star$ is the minimum value of the target function. From a practical point of view, $F^\star$ is computed using a full gradient descent algorithm run for 10^5 epochs.
- Acc_k, accuracy at iteration k which is computed as follows:

$$Acc_k = \frac{n - \frac{1}{2}\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n},$$

  where the predicted label $\hat{y}_i$ is computed as $\hat{y}_i = \text{sign}(w_k^\top x_i)$.

5.2. Numerical results

In this section, we compare the methods reported in Sections 2 and 3 and summarized in Table 2, including *RMSprop* [9] as a baseline reference method.

The first goal of these experiments is to show that the performance of the methods with line-search is at least comparable to that of the methods using a diminishing step-size. Indeed, in this case, methods with automatic step-size selection result advantageous because they do not require an initial step-size and decreasing law tuning phase. Such tuning can be very expensive or even impractical in certain big data scenarios, as it depends on numerous elements, such as the objective function, the dataset, or the mini-batch size. In this case, in fact, it is necessary to find the right combination of hyperparameters that not only ensures a fast descent of the objective function in the early epochs but also guarantees long-term convergence. This requires testing all possible combinations throughout the entire training process of the model. The use of a line-search strategy, such as the one presented in the Algorithm 1, enables automatic step-size selection at each iteration. This adaptive nature allows for a much more convenient and less costly fine-tuning of the line-search parameters. The effort is reduced to identifying the values that trigger an effective automatic step-size selection during the early stages of the process. By definition, the adaptive nature of the method ensures that the line-search dynamically adjusts to the different phases of the optimization process.

The second objective of the experiments is to assess whether the free mini-batch variant proposed in Section 3.2 is more efficient, both in terms of execution time and loss function minimization, compared to the mini-batch version proposed in [19].

To clearly highlight what has just been described, each plot compares the following numerical schemes:

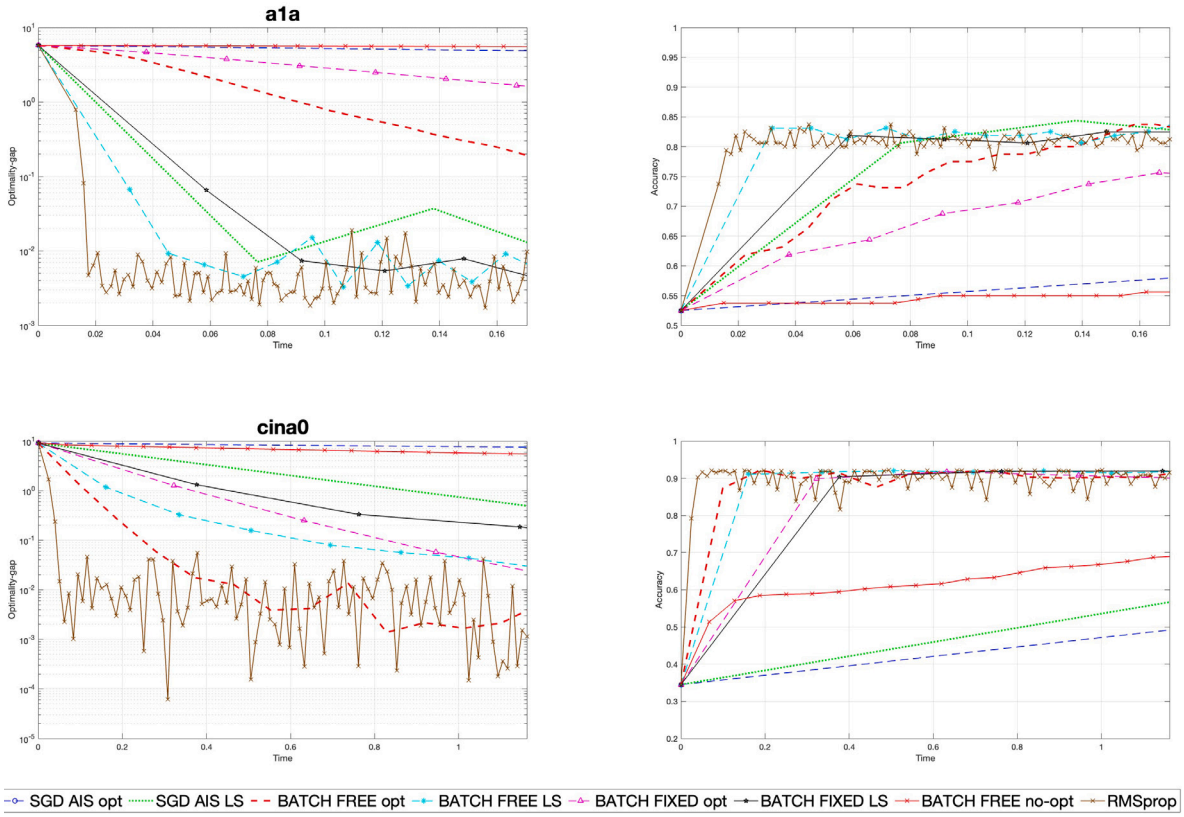

Fig. 1. Results for logistic regression.

- *Adaptive Importance Sampling methods*: the 6 methods listed in Table 2. In particular, for methods with diminishing step-size, the optimal initial step-size was selected through grid-search, as described in Section 5.2.1;
- *BATCH FREE non-optimal method (BATCH FREE no-opt)*: consider the *BATCH FREE opt* method. The same algorithm was used, but with an initial step-size η_0 equal to the optimal one multiplied by 10^{-2} to simulate a wrong choice of this hyperparameter. In this way, the strong dependence of the method on the selection of the initial step-size is highlighted. Including this method in the comparison allows us to appreciate the effectiveness of an automatic step-size selection strategy, providing an example of the effects of an incorrect yet still reasonable choice of this hyperparameter.
- *RMSprop*: a widely used optimization method based on an adaptive learning rate strategy [9], which nevertheless relies on an initial choice of a step-size value η_0. It is included here as a baseline method that does not rely on importance sampling. The step-size and decay rate were tuned via grid search to ensure fair comparison.

The analysis of accuracy and the comparison between the loss functions are conducted with respect to execution time, offering a realistic assessment of the progress of the optimization process. The time horizon considered corresponds to the time needed to complete 100 epochs with the fastest method. Furthermore, for each loss function, we report the values of loss and accuracy at the last epoch. Detailed information related to the performance over the epochs is provided in tabular form, where the loss and accuracy values at the final epoch for each method are reported.

From the results, the following evidence emerges for each comparison considered and will be further highlighted in each specific Section from 5.2.2 to 5.2.5:

- *Stochastic line-search:* first of all, we consider the comparison based on time graphs. Comparing methods in terms of time rather than epochs is a fairer approach when dealing with such heterogeneous methods in a big data scenario where memory traffic is not a negligible cost. This is necessary because we are comparing methods with and without line-search and with different batch sizes, factors that imply a different number of objective function evaluations and iterations per epoch. When analyzing the loss graphs with respect to time, two advantageous aspects of methods with line-search can be highlighted. First, they exhibit a steeper descent in the initial iterations compared to methods without line-search. Second, in the long-term behavior, their descent is comparable to that of methods without line-search but with an optimal initial step-size, or, if their performance is lower, they still outperform the "non-optimal" *BATCH FREE no-opt* method. On the other hand, when comparing based on epochs, the best-performing method appears to be *SGD-AIS*, which shows a stable descent behavior. In

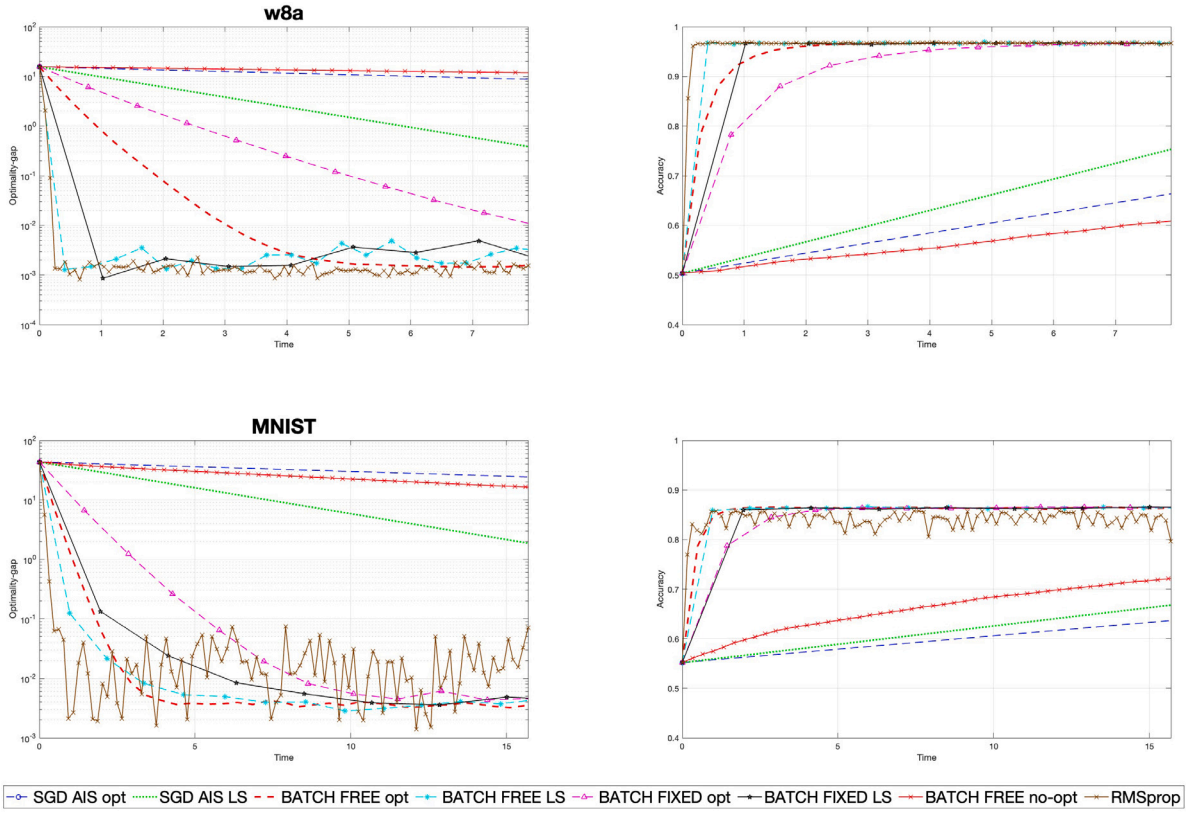

Fig. 2. Results for logistic regression.

contrast, the other methods, after the first few epochs, tend to oscillate above the minimum value without displaying a clear long-term convergence trend.

- *Free mini-batch:* the approach proposed in Section 3.2 for methods with a mini-batch size greater than one appears to be more advantageous than the one illustrated in Section 2.2 and reported in [19]. In fact, *BATCH FREE opt* methods exhibit performance that is entirely comparable to, if not better than, *BATCH FIXED opt* methods, while being more efficient in terms of execution time.
- *RMSprop* shows competitive performance in terms of execution time, reaching low loss values faster than several line-search-based methods. However, its optimization path is characterized by higher oscillations, especially in the later stages, and a fine-tuning of the initial step-size η_0 is required. Moreover, the final accuracy reached by *RMSprop* is generally lower than that achieved by mini-batch line-search methods, which exhibit more stable and consistent convergence behavior. This highlights the advantage of adaptive line-search strategies in improving robustness and long-term performance.

This section is structured as follows. In Section 5.2.1, details regarding the hyperparameters selection for each algorithm under consideration are provided. The results regarding the four different problems are presented in Subsections from 5.2.2 to 5.2.5. In particular, we show the plots of the optimality gap calculated on the training set and of the accuracy on the test set.

5.2.1. Hyperparameters details

The hyperparameters that are common to all the considered methods were set as follows: the lower and upper bounds for the step-size hyperparameter, $\underline{\alpha}$ and $\overline{\alpha}$, are fixed at 0.3 and 0.8, respectively. The number of epochs was set to 100, resulting in a maximum number of iterations $maxit = \left\lceil \frac{n}{m} \right\rceil \times 100$. The hyperparameter μ was defined as $\lambda\eta_0$, while m was chosen as $\lceil \frac{n}{100} \rceil$. For the line-search methods, we used $\gamma = 0.95$, $\beta = 0.5$, and the maximum step-size $\eta_{\max} = 1$. For computational efficiency reasons, the maximum number of backtracking iterations is set to 20.

The optimal initial step-sizes η_0 used for diminishing step-size methods are listed in Table 3. These values were empirically selected from the set

$$\{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$$

by comparing the final loss values.

Table 3
Optimal values for initial step-size η_0 in the case of decreasing step-size.

Dataset	Loss function	SGD-AIS	BATCH FREE	BATCH FIXED	RMSprop
a1a	*LR*	1e−3	1e−2	1e−2	1e−2
	SMH	1e−3	1e−1	1e−1	1e−3
	SQH	1e−3	1e−2	1e−2	1e−2
	2NN	1e−3	1e−2	1e−2	1e−3
cina0	*LR*	1e−4	1e−1	1e−1	1e−2
	SMH	1e−4	1e−2	1e−2	1e−3
	SQH	1e−4	1e−2	1e−2	1e−2
	2NN	1e−4	1e−1	1e−1	1e−3
w8a	*LR*	1e−4	1e−2	1e−2	1e−3
	SMH	1e−4	1e−2	1e−2	1e−2
	SQH	1e−4	1e−2	1e−2	1e−2
	2NN	1e−4	1e−2	1e−2	1e−3
MNIST	*LR*	1e−4	1e−1	1e−1	1e−4
	SMH	1e−4	1e−1	1e−1	1e−3
	SQH	1e−4	1e−1	1e−1	1e−4
	2NN	1e−4	1e−1	1e−1	1e-2

Table 4
Accuracy and loss value at the last epoch for logistic regression.

Method	Accuracy	Loss	Time (s)
a1a			
SGD-AIS opt	0.8063	0.4749	3.19
SGD-AIS LS	0.8187	0.4797	5.22
BATCH FREE opt	0.8313	0.4829	0.84
BATCH FREE LS	0.8125	0.4800	1.08
BATCH FIXED opt	0.8375	0.4868	2.21
BATCH FIXED LS	0.8250	0.5019	2.48
BATCH FREE no-opt	0.6250	5.1126	0.78
RMSprop	0.8125	0.4847	0.17
cina0			
SGD-AIS opt	0.9058	0.3347	234.52
SGD-AIS LS	0.9108	0.4016	296.41
BATCH FREE opt	0.9152	0.3393	8.97
BATCH FREE LS	0.9164	0.3388	17.48
BATCH FIXED opt	0.9183	0.3503	30.62
BATCH FIXED LS	0.9183	0.3517	40.29
BATCH FREE no-opt	0.9108	1.4908	8.80
RMSprop	0.9152	0.3359	1.18
w8a			
SGD-AIS opt	0.9682	0.4094	1299.45
SGD-AIS LS	0.9672	0.4240	1368.74
BATCH FREE opt	0.9668	0.4109	29.32
BATCH FREE LS	0.9676	0.4118	40.93
BATCH FIXED opt	0.9664	0.4151	78.26
BATCH FIXED LS	0.9680	0.4170	101.87
BATCH FREE no-opt	0.9664	0.4195	28.98
RMSprop	0.9668	0.4109	7.90
MNIST			
SGD-AIS opt	0.8560	0.4316	1978.01
SGD-AIS LS	0.8459	0.4588	2236.66
BATCH FREE opt	0.8641	0.4350	48.59
BATCH FREE LS	0.8639	0.4355	127.10
BATCH FIXED opt	0.8699	0.4466	149.19
BATCH FIXED LS	0.8723	0.4469	229.16
BATCH FREE no-opt	0.7941	6.5423	46.06
RMSprop	0.7969	0.5024	15.70

5.2.2. Logistic regression results

In this section, we present the results for the logistic regression loss case. In particular, Figs. 1 and 2 show the plots of the optimality gap computed for the training set and the accuracy on the test set. Table 4 completes the analysis by reporting the loss and accuracy values at the final epoch.

Table 5
Accuracy and loss value at the last epoch for smooth hinge loss.

Method	Accuracy	Loss	Time (s)
a1a			
SGD-AIS opt	0.8313	0.2473	2.89
SGD-AIS LS	0.8250	0.2724	6.42
BATCH FREE opt	0.8250	0.2498	0.74
BATCH FREE LS	0.8438	0.2575	1.05
BATCH FIXED opt	0.8375	0.2502	1.99
BATCH FIXED LS	0.8250	0.2565	2.38
BATCH FREE no-opt	0.8250	0.2503	0.69
RMSprop	0.8375	0.2712	0.25
cina0			
SGD-AIS opt	0.9214	0.1489	188.25
SGD-AIS LS	0.8983	0.2902	242.56
BATCH FREE opt	0.9233	0.1496	14.52
BATCH FREE LS	0.9264	0.1516	19.76
BATCH FIXED opt	0.9220	0.1509	28.38
BATCH FIXED LS	0.9208	0.1525	34.18
BATCH FREE no-opt	0.9220	0.1501	14.37
RMSprop	0.9189	0.1588	1.07
w8a			
SGD-AIS opt	0.9668	0.1824	1232.63
SGD-AIS LS	0.9632	0.2191	1348.24
BATCH FREE opt	0.9698	0.1843	16.82
BATCH FREE LS	0.9690	0.1846	25.75
BATCH FIXED opt	0.9723	0.1886	61.67
BATCH FIXED LS	0.9729	0.1904	77.87
BATCH FREE no-opt	0.8981	2.3810	16.92
RMSprop	0.9682	0.1843	5.15
MNIST			
SGD-AIS opt	0.8754	0.2107	2034.36
SGD-AIS LS	0.8360	0.3016	2434.32
BATCH FREE opt	0.8802	0.2120	33.22
BATCH FREE LS	0.8810	0.2122	84.86
BATCH FIXED opt	0.8853	0.2161	92.05
BATCH FIXED LS	0.8863	0.2164	149.01
BATCH FREE no-opt	0.8798	0.2123	31.45
RMSprop	0.8370	0.2532	12.78

For this first loss function, looking at the graphs and tables, we can notice that the *BATCH FREE LS* method is systematically placed between the optimal and the non-optimal diminishing step versions. Moreover, in the a1a and w8a datasets, the method obtains an even lower optimality gap than the diminishing step variants. Although *SGD-AIS* shows excellent performance at the end of the epochs, its execution times are not comparable to those of the BATCH methods, both with line-search and with diminishing step. *SGD* with line-search is the method that performs the worst, likely because the line-search condition evaluated on a single sample is not sufficient to provide a reliable step-size value.

5.2.3. Smooth hinge loss results

In this section, we present the results for the case of smooth hinge loss. In particular, Figs. 3 and 4 show the plots of the optimality gap calculated for the training set and the accuracy on the test set. Table 5 completes the analysis by reporting the loss and accuracy values at the final epoch.

For this second loss function, the graphs and tables show that the *BATCH FREE LS* method always ranks between the optimal and the non-optimal diminishing step variants. However, the improvement in terms of optimality gap compared to the diminishing step methods is mainly observed in the w8a dataset. Furthermore, we can notice that our *BATCH FREE* method shows a higher consistency compared to *BATCH FIXED*, confirming a better stability in performance.

5.2.4. Squared hinge loss results

In this section, we present the results for the case of squared hinge loss. In particular, Figs. 5 and 6 show the plots of the optimality gap calculated for the training set and the accuracy on the test set. Table 6 completes the analysis by reporting the loss and accuracy values at the final epoch.

For the third loss function, the results confirm a trend similar to the previous cases. *BATCH FREE LS* systematically falls between the optimal and the non-optimal diminishing step version. The improvement in terms of optimality gap is evident in the w8a dataset. A relevant aspect is that *BATCH FREE* continues to prove more stable than *BATCH FIXED*, ensuring more consistent performance over time

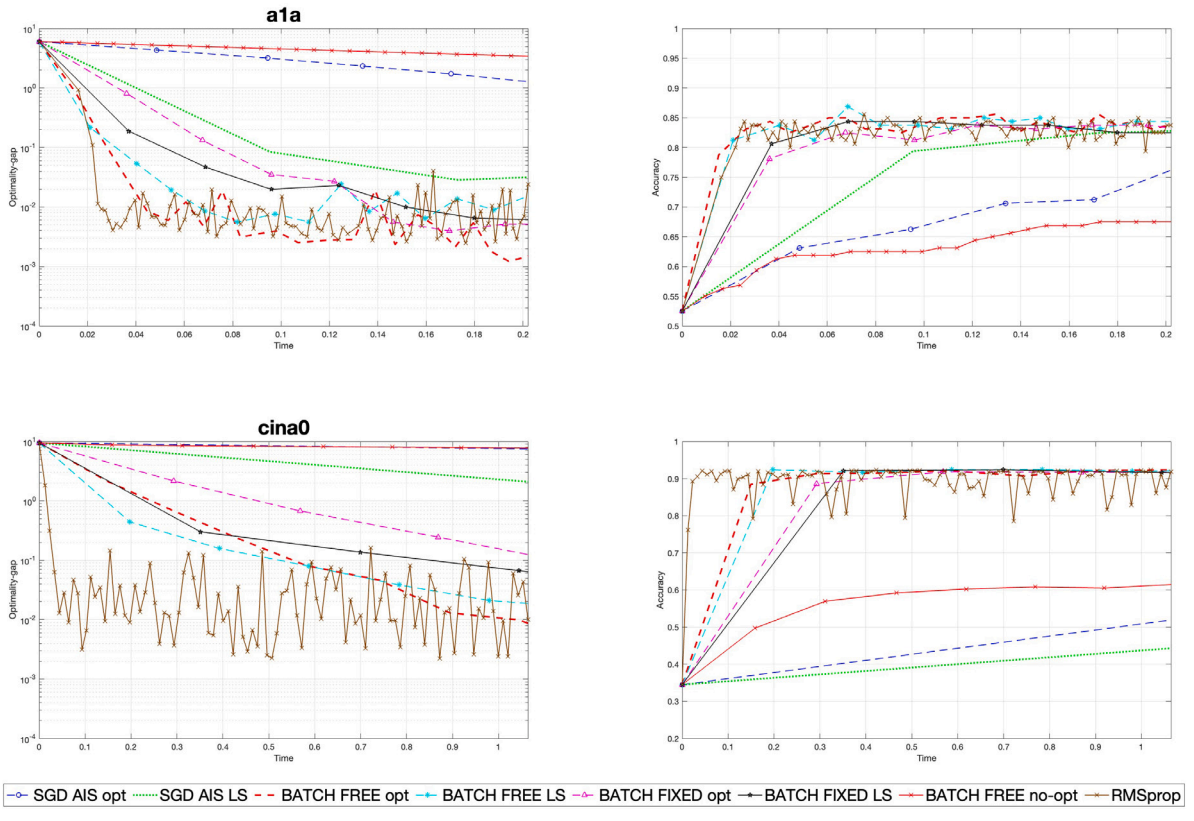

Fig. 3. Results for smooth hinge loss.

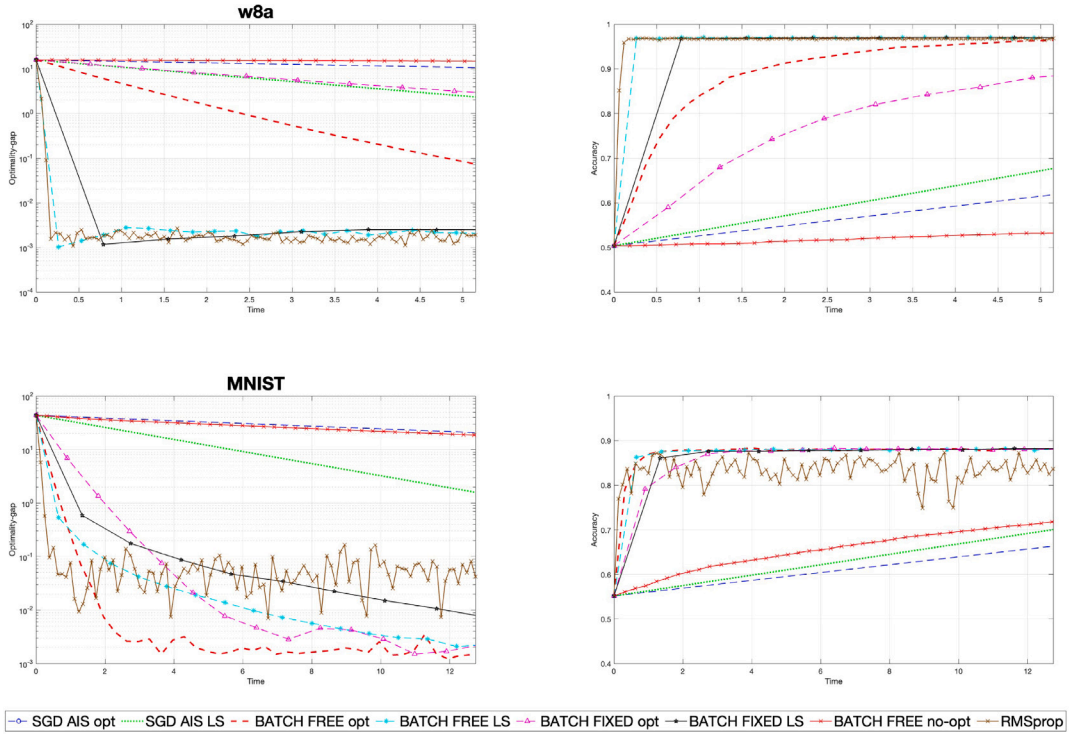

Fig. 4. Results for smooth hinge loss.

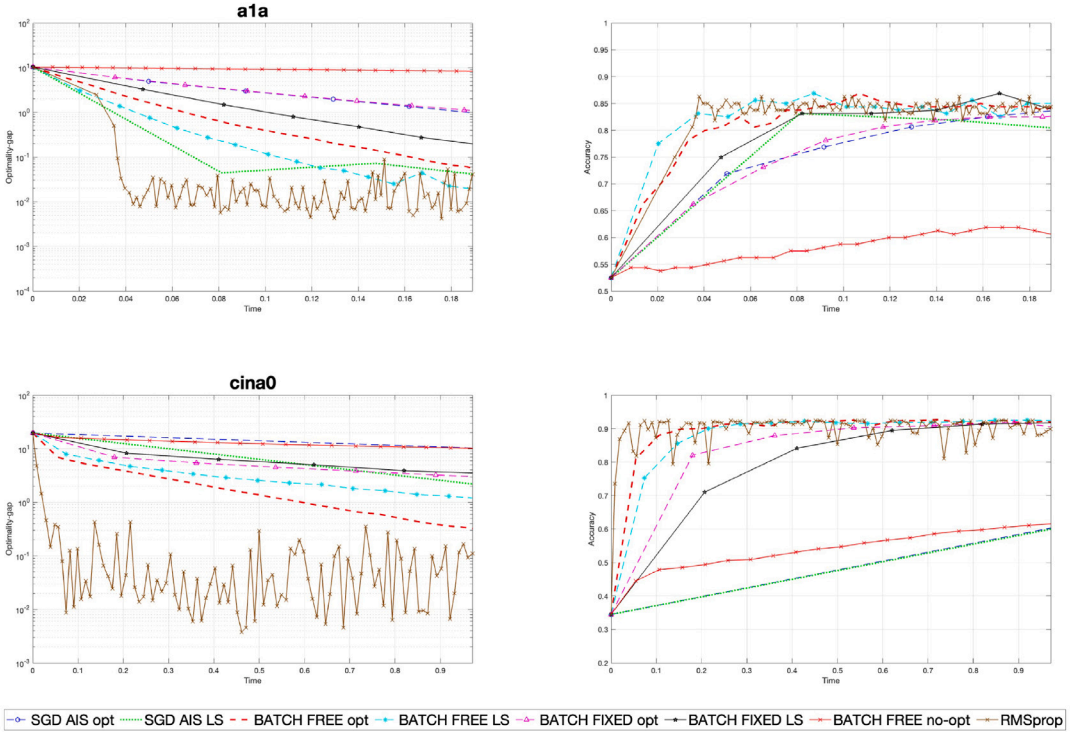

Fig. 5. Results for squared hinge loss.

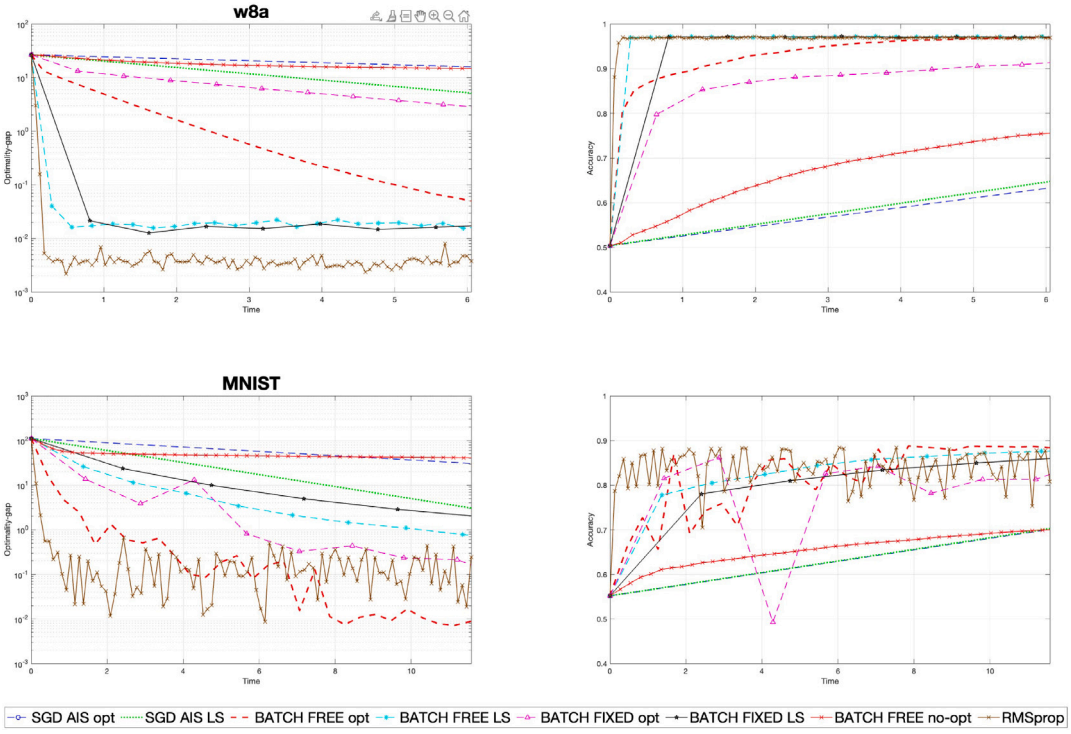

Fig. 6. Results for squared hinge loss.

Table 6
Accuracy and loss value at the last epoch for squared hinge loss.

Method	Accuracy	Loss	Time (s)
a1a			
SGD-AIS opt	0.8250	0.4830	2.96
SGD-AIS LS	0.8438	0.5271	5.14
BATCH FREE opt	0.8562	0.4850	0.68
BATCH FREE LS	0.8438	0.4973	0.98
BATCH FIXED opt	0.8313	0.4993	2.07
BATCH FIXED LS	0.8313	0.5038	2.40
BATCH FREE no-opt	0.7000	5.9364	0.66
RMSprop	0.8438	0.5236	0.20
cina0			
SGD-AIS opt	0.9226	0.3012	186.15
SGD-AIS LS	0.8921	0.4113	215.53
BATCH FREE opt	0.9258	0.3088	4.69
BATCH FREE LS	0.9245	0.3353	7.19
BATCH FIXED opt	0.9233	0.3192	18.16
BATCH FIXED LS	0.9208	0.3485	20.87
BATCH FREE no-opt	0.8428	6.5432	5.11
RMSprop	0.8989	0.4109	0.98
w8a			
SGD-AIS opt	0.9701	0.3300	1857.99
SGD-AIS LS	0.9660	0.3999	2025.91
BATCH FREE opt	0.9709	0.3471	15.57
BATCH FREE LS	0.9705	0.3451	28.14
BATCH FIXED opt	0.9731	0.3612	63.31
BATCH FIXED LS	0.9725	0.3614	80.64
BATCH FREE no-opt	0.8265	12.7540	15.82
RMSprop	0.9692	0.3337	6.05
MNIST			
SGD-AIS opt	0.8840	0.4135	2005.39
SGD-AIS LS	0.8136	0.5938	2177.54
BATCH FREE opt	0.8837	0.4199	41.25
BATCH FREE LS	0.8847	0.4218	158.60
BATCH FIXED opt	0.8902	0.4379	139.72
BATCH FIXED LS	0.8897	0.4377	264.24
BATCH FREE no-opt	0.7428	35.2218	41.92
RMSprop	0.8079	0.6560	11.58

5.2.5. 2-Layers Neural Network results

In this section, we present the results for the case of 2-layers Neural Network. In particular, Figs. 7 and 8 show the plots of the optimality gap calculated for the training set and the accuracy on the test set. Table 7 completes the analysis by reporting the loss and accuracy values at the final epoch.

For the fourth loss function, the analysis of the results shows that *BATCH FREE LS* maintains an intermediate behavior between the variants with diminishing step, but without clearly detaching itself. The w8a dataset continues to be the one where the greatest advantages in terms of optimality gap and accuracy are observed. One aspect is that *BATCH FREE* proves to be more reliable than *BATCH FIXED*, suggesting greater robustness in convergence.

6. Conclusions

In this work, we considered the SGD-AIS algorithm introduced in [19] and proposed two possible extensions related to automatic step-size selection and mini-batch construction. For the first case, we introduced a methodology based on a stochastic Armijo-type line-search instead of the decreasing step-size rule (3). We also conducted a theoretical convergence analysis for this case. The second proposal presents a more efficient and flexible approach to mini-batch stochastic gradient computation.

The experiments show that using line-search achieves performance comparable to algorithms with a decreasing step-size while avoiding the costly selection of the initial step-size. Additionally, the more flexible approach to mini-batch construction proves both effective and computationally more efficient than the previously proposed method. Future research can evolve in two directions. The first focuses on improving the long-term descent behavior of the objective function. In particular, methods with line-search tend to be very fast in the initial epochs but then oscillate around a value close to the optimum. To eliminate this behavior, Dynamic Sampling strategies could be applied. These strategies are now feasible due to the flexibility introduced by our proposed mini-batch construction method. The second aspect concerns a deeper convergence analysis, which currently applies to the stochastic gradient computed on a single sampled index. A natural extension would be to leverage the formulation of our mini-batch method to establish a similar convergence result for the case where the stochastic gradient is computed over multiple sampled indices.

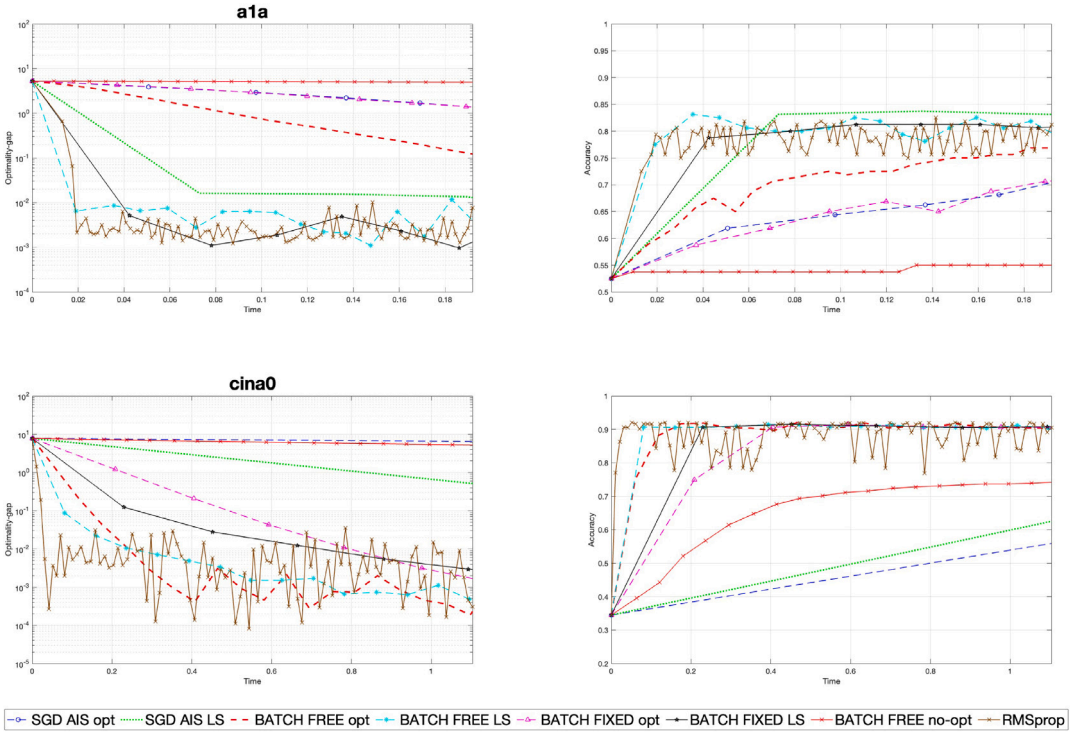

Fig. 7. Results for 2-layers Neural Network.

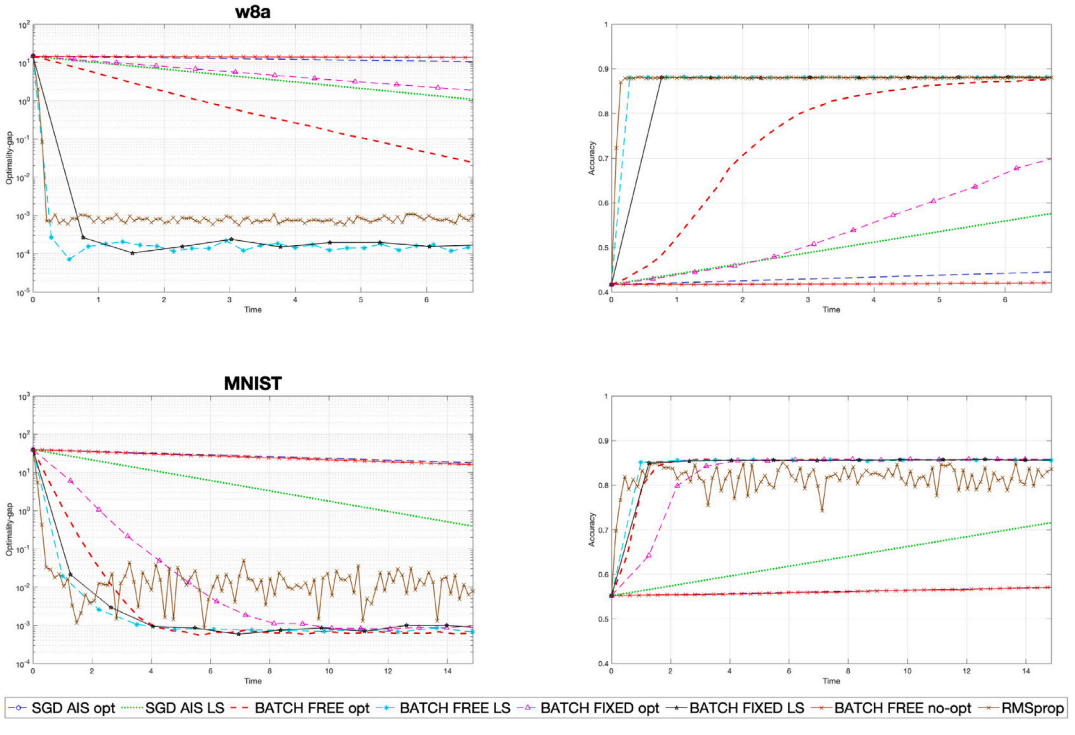

Fig. 8. Results for 2-layers Neural Network.

Table 7
Accuracy and loss value at the last epoch for 2-layers Neural Network.

Method	Accuracy	Loss	Time (s)
a1a			
SGD-AIS opt	0.8989	0.1207	3.09
SGD-AIS LS	0.7729	0.2006	4.91
BATCH FREE opt	0.9058	0.1210	0.77
BATCH FREE LS	0.9052	0.1210	0.95
BATCH FIXED opt	0.9170	0.1237	2.20
BATCH FIXED LS	0.9177	0.1234	2.41
BATCH FREE no-opt	0.7318	1.3574	0.76
RMSprop	0.8125	0.1210	0.20
cina0			
SGD-AIS opt	0.8816	0.1532	193.63
SGD-AIS LS	0.8790	0.1621	219.75
BATCH FREE opt	0.8810	0.1533	5.79
BATCH FREE LS	0.8808	0.1533	7.78
BATCH FIXED opt	0.8796	0.1537	19.51
BATCH FIXED LS	0.8798	0.1540	21.59
BATCH FREE no-opt	0.4291	12.0775	5.73
RMSprop	0.9052	0.1542	1.11
w8a			
SGD-AIS opt	0.8000	0.1709	1790.20
SGD-AIS LS	0.8250	0.1890	1913.83
BATCH FREE opt	0.8063	0.1730	17.56
BATCH FREE LS	0.8125	0.1742	26.64
BATCH FIXED opt	0.8125	0.1747	60.88
BATCH FIXED LS	0.8250	0.1814	74.61
BATCH FREE no-opt	0.5875	4.4802	17.61
textitRMSprop	0.8802	0.1785	6.71
MNIST			
SGD-AIS opt	0.8493	0.1567	2191.12
SGD-AIS LS	0.8422	0.1714	2385.76
BATCH FREE opt	0.8567	0.1573	50.06
BATCH FREE LS	0.8559	0.1573	122.97
BATCH FIXED opt	0.8638	0.1598	151.86
BATCH FIXED LS	0.8636	0.1602	423.66
BATCH FREE no-opt	0.6527	6.1124	50.98
RMSprop	0.8355	0.1645	14.86

Acknowledgments

This work was partially supported by Italian Ministry of University and Research (MIUR), PRIN 2022 Project: Numerical Optimization with Adaptive Accuracy and Applications to Machine Learning, Grant no. 2022N3ZNAX, PNRR PRIN 2022 Project: A multidisciplinary approach to evaluate ecosystems resilience under climate change, Grant no. P2022WC2ZZ, and by Istituto Nazionale di Alta Matematica - Gruppo Nazionale per il Calcolo Scientifico (INdAM-GNCS), by MIUR. This work was also supported by the Gruppo Nazionale per il Calcolo Scientifico (GNCS-INdAM). The publication was created with the co-financing of the European Union-FSE-REACT-EU, PON Research and Innovation 2014–2020 DM1062/2021.

Data availability

Data will be made available on request.

References

[1] F.E. Curtis, K. Scheinberg, Optimization methods for supervised machine learning: From linear models to deep learning, 2017, arXiv:1706.10207.
[2] L. Bottou, F.E. Curtis, J. Nocedal, Optimization methods for large-scale machine learning, SIAM Rev. 60 (2) (2018) 223–311.
[3] L. Bottou, Online algorithms and stochastic approximations, in: D. Saad (Ed.), Online Learning and Neural Networks, Cambridge University Press, Cambridge, UK, 1998, Revised, 2018.
[4] G. Garrigos, R.M. Gower, Handbook of convergence theorems for (stochastic) gradient methods, 2024, arXiv:2301.11235.
[5] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Stat. 22 (3) (1951) 400–407.
[6] S.J. Reddi, A. Hefny, S. Sra, B. Póczós, A. Smola, Stochastic variance reduction for nonconvex optimization, in: ICML 2016, 2016, pp. 314–323.
[7] A. Defazio, F. Bach, S. Lacoste-Julien, SAGA: a fast incremental gradient method with support for non-strongly convex composite objectives, in: NIPS 2014, 2014, pp. 1646–1654.

[8] J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, J. Mach. Learn. Res. (ISSN: 1532-4435) 12 (null) (2011) 2121–2159.
[9] G. Hinton, Lecture 6.5—RMSprop: Divide the gradient by a running average of its recent magnitude, 2012, Coursera: Neural Networks for Machine Learning, https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.
[10] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.
[11] R. Bollapragada, R. Byrd, J. Nocedal, Adaptive sampling strategies for stochastic optimization, SIAM J. Optim. 28 (4) (2018) 3312–3343.
[12] G. Franchini, F. Porta, V. Ruggiero, I. Trombini, L. Zanni, A stochastic gradient method with variance control and variable learning rate for deep learning, J. Comput. Appl. Math. 451 (2024).
[13] G. Franchini, F. Porta, V. Ruggiero, I. Trombini, A line search based proximal stochastic gradient algorithm with dynamical variance reduction, J. Sci. Comput. 94 (23) (2023).
[14] S. Bellavia, N. Krejić, N.K. Jerinkić, M. Raydan, SLiSeS: subsampled line search spectral gradient method for finite sums, Optim. Methods Softw. (2024) 1–26.
[15] N. Krklec Jerinkić, V. Ruggiero, I. Trombini, Spectral stochastic gradient method with additional sampling for finite and infinite sums, Comput. Optim. Appl. 91 (2) (2025) 717–758.
[16] R.H. Byrd, G.M. Chin, J. Nocedal, Y. Wu, Sample size selection in optimization methods for machine learning, Math. Program. 134 (1) (2012) 127–155.
[17] S. Horváth, P. Richtárik, Nonconvex variance reduced optimization with arbitrary sampling, in: International Conference on Machine Learning, 2019, pp. 2781–2789.
[18] L. Xiao, T. Zhang, A proximal stochastic gradient method with progressive variance reduction, SIAM J. Optim. 24 (4) (2014) 2057–2075.
[19] H. Liu, X. Wang, J. Li, A.M.-C. So, Low-cost Lipschitz-independent adaptive importance sampling of stochastic gradients, in: ICPR 2020, 2021, pp. 2150–2157.
[20] A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM J. Imaging Sci. 2 (1) (2009) 183–202.
[21] M. Fazlyab, A. Robey, H. Hassani, M. Morari, G. Pappas, Efficient and accurate estimation of Lipschitz constants for deep neural networks, in: Advances in Neural Information Processing Systems, vol. 32, 2019.
[22] S. Vaswani, A. Mishkin, I. Laradji, M. Schmidt, G. Gidel, S. Lacoste-Julien, Painless stochastic gradient: interpolation, line-search, and convergence rates, in: NIPS 2019, 2019.
[23] M. Mahsereci, P. Hennig, Probabilistic line searches for stochastic optimization, in: Advances in Neural Information Processing Systems, vol. 28, 2015.
[24] R. Huang, Y. Qin, K. Liu, G. Yuan, Biased stochastic conjugate gradient algorithm with adaptive step size for nonconvex problems, Expert Syst. Appl. 238 (2024) 121556.
[25] C. Ouyang, C. Lu, X. Zhao, R. Huang, G. Yuan, Y. Jiang, Stochastic three-term conjugate gradient method with variance technique for non-convex learning, Stat. Comput. 34 (3) (2024) 107.
[26] C. Ouyang, A. Jian, X. Zhao, G. Yuan, Difference-enhanced adaptive momentum methods for non-convex stochastic optimization in image classification, Digit. Signal Process. 161 (2025) 105118.
[27] Z. Mo, C. Ouyang, H. Pham, G. Yuan, A stochastic recursive gradient algorithm with inertial extrapolation for non-convex problems and machine learning, Int. J. Mach. Learn. Cybern. (2025).
[28] T. Sun, H. Ling, Z. Shi, D. Li, B. Wang, Training deep neural networks with adaptive momentum inspired by the quadratic optimization, 2021, arXiv preprint arXiv:2110.09057.
[29] L. Armijo, Minimization of functions having Lipschitz continuous first partial derivatives, Pacific J. Math. 16 (1966) 1–3.
[30] G. Franchini, F. Porta, V. Ruggiero, I. Trombini, L. Zanni, Learning rate selection in stochastic gradient methods based on line search strategies, Appl. Math. Sci. Eng. 31 (1) (2023).
[31] P. Zhao, T. Zhang, Stochastic optimization with importance sampling for regularized loss minimization, in: PMLR 2015, 2015, pp. 1–9.
[32] D.P. Bertsekas, Nonlinear programming, J. Oper. Res. Soc. 48 (3) (1997) 334–334.
[33] D. Bertsekas, Convex Optimization Theory, Athena Scientific, Belmont, Massachussets, 2009.
[34] Libsvm, https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/.
[35] Cha Learn, https://www.causality.inf.ethz.ch/challenge.php?page=datasets.