Stochastic Proximal Linear Method for Structured Non-convex Problems

Tamir Hazan* Shoham Sabach† Sergey Voldman‡

February 4, 2020

Abstract

In this work, motivated by the challenging task of learning a deep neural network, we consider optimization problems which consist of minimizing a finite-sum of non-convex and non-smooth functions, where the non-smoothness appears as the maximum of non-convex functions with Lipschitz continuous gradient. Due to the large size of the sum, in practice, we focus here on stochastic first-order methods and propose the Stochastic Proximal Linear Method (SPLM) that is based on minimizing an appropriate majoraizer at each iteration and is guaranteed to almost surely converge a critical point of the objective function, where we also proves its convergence rate in finding critical points.

2010 Mathematics Subject Classification: Primary 90C25; Secondary 26B25, 49M27, 52A41, 65K05.
Keywords: Non-convex non-smooth minimization, proximal-gradient algorithms, stochastic, convergence analysis.

1 Introduction

In recent years deep neural networks drive much of the research in machine learning applications, from computer vision, natural language processing, to gaming. Examples include image convolutional neural networks that dominate the state-of-the-art in computer vision, recurrent neural networks that dominate the state-of-the-art in speech recognition and deep Q-networks that achieved human expert level in gaming.

Learning a deep neural network amounts to optimize a sum of non-convex non-smooth functions. While current theory does not provide convergence guarantees for optimizing deep learners, stochastic gradient descent variants turn to be very effective in machine learning. In this work, motivated by the structure of deep learning optimization problems, which is the sum of non-convex and non-smooth functions, we focus on a special case, where the non-smoothness appears as the maximum of non-convex functions with Lipschitz continuous gradient. We develop the Stochastic Proximal Linear Method (SPLM), which is desgined to exploit this structure via simple computational steps. In addition, we prove that SPLM is guaranteed to almost surely reach a critical point of the objective function at hand. For this purpose we majorize the objective function and show that the majorization function has the same critical points as the learning objective. We support our convergence analysis with an additional result showing that SPLM has an $O(m^2/\varepsilon^2)$

*Faculty of Industrial Engineering, The Technion, Haifa, 32000, Israel. E-mail: tamir.hazan@technion.ac.il
†Faculty of Industrial Engineering, The Technion, Haifa, 32000, Israel. E-mail: ssabach@ie.technion.ac.il.
‡Faculty of Industrial Engineering, The Technion, Haifa, 32000, Israel. E-mail: sergeyv@campus.technion.ac.il

convergence rate, where ε is the desired accuracy and m is the number of involved functions in the finite sum. To the best of our knowledge, this is the only stochastic gradient-based algorithm that is provably convergent for non-convex and non-smooth optimization that is suitable for deep learning networks.

1.1 Related work

Optimization algorithms have gained a lot of attraction in the last decades, especially in relation to machine learning applications, for a review paper we point the readers to [6]. In the last five years , with the emergence of deep neural networks, the sub-field of non-convex optimization attracted worldwide research activities conducted at several directions (for a taste of applications, see [28, 15]). A main direction of research, that is very common, revolves around stochastic optimization methods. Such methods gained and intensive popularty due to the huge size of the problems to be solved in this domain of applications, that usually preclude applying deterministic methods. This line of research was initiated with the development of the Stochatic Gradient Descent (SGD) method in the seminal work of Robbins and Monro [21], which paved the way to many contributions mostly in the convex setting (see [6] and the references therein). On the other hand, in the non-convex setting, the situation is much more difficult and the research is in its infancy.

Motivated by the application at hand, especially the training of neural networks, the finite-sum model (properly defined below) becomes a source for most of the research in this area. We will highlight a few directions of fruitful research in this area. Algorithms which use variance reduction technique, first in the convex setting [24, 13, 10] and later on in the non-convex setting [18]. In [9], a block coordinate stochastic proximal gradient, was established. Another direction is of methods which are able to escape from saddle points and guaranteed to converge to second order stationary points. For instance, [26] show that adaptive methods, such as Adam and RMSProp, are faster than SGD. In [8] the authors provide provide an accelerated method with similar guarantees. The work [12] showed that a noise-injected version of the SGD method converges to a local minima instead of critical points, as long as the underlying non-convex function is strict-saddle. The work [16] shows that gradient descent, starting from a random point, almost surely converges to a local minimum of a strict-saddle function. Some works on second order methods that improve the convergence rate of first order method include, e.g., [1, 7, 20].

A common assumption to all these works and most of the research in these directions, is that the involved functions in the finite-sum are smooth and even with Lipschitz continuous gradient. This is a very restrictive assumption which limits the applicability of the proposed methods in many practical situations. In this work we depart from this assumption and study a finite-sum model, which allow non-smoothness to these functions using the max-function, as we describe next.

1.2 Our setting and optimization model

In contrast to the works mentioned above, our work considers the composition of the max-function with smooth and non-convex functions. Such compositions are instrumental in building deep learners, either by using the ReLU transfer function (mathematically defined by $f(t) = \max\{0, t\}$) or using max-pooling (a common down sampling operation that takes the maximal sample over certain part of the data). To be precise, in this paper we are focusing on the finite-sum model of the following form

$$\min_{w\in\mathbb{R}^p} \left\{ g(w) := \frac{1}{m}\sum_{i=1}^{m} g_i(w) \right\},$$

where each function $g_i : \mathbb{R}^p \to \mathbb{R}$, $i = 1, 2, \ldots, m$, is of the following form

$$g_i(w) = \max_{\hat{y} \in Y} g_{i,\hat{y}}(w),$$

and Y is a certain index set (see precise definition in the following section) and $g_{i,\hat{y}}$ are assumed to be continuously differentiable functions with Lipschitz continuous gradient.

The objective of the present work is to tackle this non-convex and non-smooth sum model by exploiting structure features which will be in the basis of the proposed method. Our work provides a stochastic version of the Proximal Linear Method (PLM) recently studied by Bolte and Pauwels [5], which is a deterministic algorithm that is proven to globally converges to critical points of g where $m = 1$. The stochastic extension of PLM, which we denote by SPLM, allows one addressing applications, which consist of large sum, like in deep learning, where the gradient of each function in the finite-sum has a significant computational and memory overhead.

2 Background

In deep learning, the basic learner unit is a linear classifier. In multi-class setting, a data point $x \in \mathbb{R}^d$ is classified to a class $y \in Y$ by its parameters $W \in \mathbb{R}^{|Y| \times d}$ according to the highest scoring value of Wx. Therefore, in order to make these notations clear, when we write $(Wx)_y$ we mean the y-coordinate (can be thought as an index) of the vector Wx.

Given a training data set $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, which consists of data points x and their labels $y \in Y$, a linear classifier is learned by minimizing a loss function $\ell(Wx, y)$ that penalizes misclassified points. The training loss minimization problem takes the following form

$$\min_W \frac{1}{m} \sum_{(x,y) \in S} \ell(Wx, y).$$

Two classical examples are the log-loss function $\ell(Wx, y) = \log(\sum_{\hat{y}} \exp((Wx)_{\hat{y}})) - (Wx)_y$, which is used in logistic regression, and the hinge loss $\ell(Wx, y) = \max_{\hat{y}} \{\hat{\ell}(\hat{y}, y) + (Wx)_{\hat{y}} - (Wx)_y\}$, which is used in support vector machines, when $\hat{\ell}(\hat{y}, y)$ is a label loss, i.e., $\hat{\ell}(\hat{y}, y) \geq 0$ and $\hat{\ell}(y, y) = 0$. Both these loss functions are convex and thus can beminimized by (Sub)Gradient-based Descent Methods, which are guaranteed to reach the global optimum of the training problem. Due to the fact that contemporary data sets consist of significant amount of data points, a major concern is to relax the memory constraints of the applied optimization algorithm, which is usually cured by appling Stochastic Gradient-based Descent Methods to a single data point or a small batch of data points. The expected value of the stochastic (sub)gradient is the true (sub)gradient and convergence to the global optimum of the convex problem at hand can be derived (see, for instance, [6]).

Deep neural networks consist of array of linear classifiers. For notational convenience we describe fully connected networks with l-layers. The neurons in the k-th layer are linear classifiers whose parameters are the rows in the matrix W^k. The input vector of the k-th layer, denoted by $x^k \in \mathbb{R}^d$, is a function of the parameters of the previous layers, namely $x^k\left(W^0, W^1, \ldots, W^{k-1}\right)$. The entries of x^k are computed from the response of its preceding layer classifiers $W^{k-1}x^{k-1}$, followed by a transfer function, that is, $x^k = f\left(W^{k-1}x^{k-1}\right)$. The function $f(\cdot)$ introduces non-linearities to the process. Consequently, the deep learning optimization objective is non-convex. Moreover, in most architectures, the transfer function is the non-smooth ReLU function $f(t) = \max\{0, t\}$ and

consequently the learning objective is non-smooth as well as non-convex. Therefore, learning a network with l-layers, amounts to the following optimization problem

$$\min_{w} \frac{1}{m} \sum_{(x,y)\in S} \ell\left(W^l x^l, y\right),$$

where $x^k = f\left(W^{k-1}x^{k-1}\right)$, $k \in \{1, 2, \ldots, l\}$, and w is the concatenation of all neurons in the network $\left(W^l, W^{l-1}, \ldots, W^0\right)$.

Unfortunately, contemporary algorithms, when applied to these architectures, are not guaranteed to converge to a critical point of the corresponding objective function. In the following we develop an almost surely convergent algorithm for this non-smooth and non-convex learning objective. Our non-smooth component appears in the loss function, e.g., the hinge loss. The transfer functions we are using are the smooth softplus functions $f(t) = \log(1 + \exp(t))$, which smoothly approximate the ReLU function. Therefore, in this case the situation exactly fits into our general model (see Section 1.2), and can be written as follows

$$g(w) = \frac{1}{m}\sum_{i=1}^{m} g_i(w) = \frac{1}{m}\sum_{i=1}^{m} \max_{\hat{y}\in Y} g_{i,\hat{y}}(w) = \frac{1}{m}\sum_{i=1}^{m} \max_{\hat{y}} \left\{\hat{\ell}(\hat{y}, y) + \left(W^l x_i^l\right)_{\hat{y}} - \left(W^l x_i^l\right)_{y_i}\right\}. \quad (2.1)$$

We use the shorthand $g_i(w)$ to denote the loss of the i-th training example to avoid notational overhead. Also, we note that $\left(W^l x^l\right)_y$ has a Lipschitz continuous gradient and its Lipschitz constant depends on the norm of x and W.

The main difficulty with our non-smooth component, which is introduced by the max-function, comes from the non-convexity of $g_i(w)$. Clearly, if $W^l x^l$ would have been a convex function of $w = \left(W^l, W^{l-1}, \ldots, W^0\right)$, then the function $g(w)$ would have been convex as well and its sub-differential would have been the subset of vectors d for which $g(w) - g(u) - \langle d, w - u\rangle \geq 0$. Since deep learners introduce non-convexity, we deal with the following extension of sub-differential to non-convex functions (cf. [23, Definition 8.3]).

Definition 2.1 (Subdifferentials)**.** Let $g : \mathbb{R}^p \to (-\infty, +\infty]$ be a proper and lower semicontinuous function.

(i) For a given $w \in \operatorname{dom} h$, the *Fréchet subdifferential* of g at w, written $\widehat{\partial} g(w)$, is the set of all vectors $d \in \mathbb{R}^p$ which satisfy

$$\liminf_{\substack{u \neq w \\ u \to w}} \frac{g(u) - g(w) - \langle d, u - w\rangle}{\|u - w\|} \geq 0.$$

When $w \notin \operatorname{dom} g$, we set $\widehat{\partial} g(w) = \emptyset$.

(ii) The *limiting-subdifferential*, or simply the subdifferential, of g at $w \in \mathbb{R}^p$, written $\partial g(w)$, is defined through the following closure process

$$\partial g(w) := \left\{d \in \mathbb{R}^p : \exists\, w^k \to w, g\left(w^k\right) \to g(w) \text{ and } d^k \in \widehat{\partial} g\left(w^k\right) \to d \text{ as } k \to \infty\right\}.$$

Both sub-differential sets $\widehat{\partial} g(w)$ and $\partial g(w)$ are closed sets for any $w \in \mathbb{R}^p$, while the set $\widehat{\partial} g(w)$ is also convex (see [23, Theorem 8.6]). We also have that $\widehat{\partial} g(w) \subset \partial g(w)$ for any $w \in \mathbb{R}^p$.

3 Algorithm and Convergence Analysis

In this section we are focusing on the optimization problem defined in (2.1). Specifically, minimizing

$$g(w) = \frac{1}{m}\sum_{i=1}^{m} g_i(w) = \frac{1}{m}\sum_{i=1}^{m} \max_{\hat{y}\in Y} g_{i,\hat{y}}(w).$$

Our main objective is to devise a stochastic algorithm to tackle the minimization of $g(w)$ which can handle large scale instances (meaning large m). Motivated by the recent work of Bolte and Pauwels [5] (see also [2, 17, 19]), in order to achieve our main goal we propose a stochastic version of the Prox Linear Method, which is recorded now.

Stochastic Prox-Linear Method – SPLM

1. Initialization: start with any $w^0 \in \mathbb{R}^p$.
2. For each $t = 0, 1, \ldots$

   Choose a positive definite matrix M, choose randomly an index $i \in \{1, 2, \ldots, m\}$ uniformly at random and compute

   $$w^{t+1} = \operatorname{argmin}_{w\in\mathbb{R}^p} \left\{ \max_{\hat{y}\in Y} \left\{ h_{i,\hat{y}}\left(w, w^t\right)\right\} + \frac{1}{2}\left\|w - w^t\right\|_M^2 \right\}, \quad (3.1)$$

   where $h_{i,\hat{y}}\left(w, w^t\right) = g_{i,\hat{y}}\left(w^t\right) + \left\langle \nabla g_{i,\hat{y}}\left(w^t\right), w - w^t\right\rangle$.

We recall that the weighted norm is defined by $\|w\|_M := \sqrt{w^T M w}$, where M is a positive definite matrix, which means that $M \succeq \beta I$ for some $\beta > 0$.

The random index i as well as the matrix M may differ in each iteration and should be noted by the iteration index, namely M_t. Instead, we omit the iteration index and use M for clarity. Interestingly, the SPLM algorithm is a generalization of the SGD Method to linear classifiers, indeed when applied to support vector machines, the SPLM algorithm reduces to the primal estimated sub-gradient solver (Pegasos) [25].

The main idea of SPLM (as of the deterministic version) is to beneficially exploit the structure of the objective function $g(\cdot)$ by replacing each non-convex inner function $g_{i,\hat{y}}(\cdot)$ with its linear approximation around given point $\bar{w}$, which is denoted in the algorithm by $h_{i,\hat{y}}(w, \bar{w})$, and then we also add a proximal quadratic term. The idea of linearizing smooth functions and adding proximal terms stands in the basis of many optimization methods, such as proximal gradient (see [27] for a recent review paper).

Before providing the convergence analysis, we would like to say few words about how to compute an iteration of SPLM. Since we replace the function $g_{i,\hat{y}}(\cdot)$ by its linear approximation $h_{i,\hat{y}}(\cdot,\cdot)$, inside the "max", the obtained subproblem of step (3.1), consists of an unconstrained minimization of the sum of a strongly convex function and a non-smooth and convex function that can not be explicilty solved. However, fortenatly, this problem fits exactly to the model that was studied recently in the paper of Beck and Teboulle [3], where a Fast Dual-Based Proximal Gradient (FDPG) Method was proposed and analyzed. Therefore, a solution to this subproblem, at each iteration, will be approximated by applying the FDPG Method (see more details in Section 4).

3.1 Mathematical Toolbox

The convergence analysis of the SPLM method will be developed in the following sub-section. To this end we will need some technical results on the involved functions in our optimization model and algorithm. We begin with a result regarding the subdifferential of the non-convex and non-smooth objective function $g(\cdot)$.

Lemma 3.1. *Assume that $g_{i,\hat{y}} : \mathbb{R}^p \to \mathbb{R}$ are continuously differentiable functions. Then, for all $w \in \mathbb{R}^p$, we have*

$$\frac{1}{m}\sum_{i=1}^{m} \partial g_i(w) = \partial g(w).$$

Proof. We first define a function $\varphi : \mathbb{R}^{mn} \to (-\infty, +\infty]$ by

$$\varphi(z) = \frac{1}{m}\sum_{i=1}^{m} \varphi_i(z_i),$$

where $\varphi_i(z_i) = \max_{\hat{y}\in Y} z_{i,\hat{y}}$ and $z_i \in \mathbb{R}^n$. In addition, we define

$$G(x) = [G_1(x), G_2(x), \ldots, G_m(x)], \quad \text{where} \quad G_i(x) = [g_{i1}(x), g_{i2}(x), \ldots, g_{in}(x)]^T,$$

where $n = |Y|$. Then, it follows that $g(x) = \varphi(G(x))$. By using the chain rule (see [23, Theorem 10.6]) and since $\varphi(\cdot)$ is a convex function, we obtain that

$$\partial g(x) = \nabla G(x)^T \partial\varphi(G(x)) = \frac{1}{m}\nabla G(x)^T \left(\partial\varphi_1(G_1(x)) \times \partial\varphi_2(G_2(x)) \times \cdots \times \partial\varphi_m(G_m(x))\right),$$

where the last equality follows from the structures of $\varphi(\cdot)$ and G (see [23, Proposition 10.5]). Therefore

$$\partial g(x) = \frac{1}{m}\sum_{i=1}^{m} \nabla G_i(x)^T \partial\varphi_i(G_i(x)) = \frac{1}{m}\sum_{i=1}^{m} \partial g_i(x),$$

which yields the desired result. □

The above proposition holds trivially for smooth functions, since the subdifferential of each g_i, $i = 1, 2, \ldots, m$, contains only the gradient. For convex functions, the subdifferential may contain more than one vector, nevertheless the above statement holds true (under classic regularity assumptions) see, for example, [22]. Unfortunately, the above proposition does not hold for any non-convex and non-smooth function, since we are dealing with the more general notation of subdifferential (see Definition 2.1). However, it holds in our setting since we exploit the convexity of the "max" function and the differentiability of the inner functions.

Next, we use a standard descent lemma for functions with Lipschitz continuous gradient to show that in our non-smooth setting, which consists of a maximum of functions with Lipschitz continuous gradient, we can also obtain an upper majoraizer (which is not quadratic).

Lemma 3.2. *Assume that $g_{i,\hat{y}} : \mathbb{R}^p \to \mathbb{R}$ are continuously differentiable functions with $\nabla g_{i,\hat{y}}$ assumed $L_{i,\hat{y}}$-Lipschitz continuous. Then*

$$g(u) \le g(v) + \frac{1}{m}\sum_{i=1}^{m} \max_{\hat{y}\in Y} \langle \nabla g_{i,\hat{y}}(v), u - v\rangle + \frac{L}{2}\|u - v\|^2, \quad \forall\, u, v \in \mathbb{R}^p, \tag{3.2}$$

where $L = \frac{1}{m}\sum_{i=1}^{m} \max_{\hat{y}\in Y} L_{i,\hat{y}}$.

Proof. Let $1 \leq i \leq m$ and $\hat{y} \in Y$. Due to the Lipschitz property of $\nabla g_{i,\hat{y}}$ we have from the well-known Descent Lemma (see [4]) and the definition of g (see (2.1)) that

$$g_{i,\hat{y}}(u) \leq g_{i,\hat{y}}(v) + \langle \nabla g_{i,\hat{y}}(v), u - v \rangle + \frac{L_{i,\hat{y}}}{2} \|u - v\|^2. \tag{3.3}$$

Maximizing (3.3) over all $\hat{y} \in Y$, using the definition of g_i (see (2.1)) and splitting the max term on the right-hand side yields

$$g_i(u) \leq g_i(v) + \max_{\hat{y} \in Y} \langle \nabla g_{i,\hat{y}}(v), u - v \rangle + \max_{\hat{y} \in Y} \frac{L_{i,\hat{y}}}{2} \|u - v\|^2. \tag{3.4}$$

Summing (3.4) over $i = 1, 2, \ldots, m$ and dividing by m, yields the result. □

The idea behind the main step of SPLM, which is based on the observation obtained in Lemma 3.2, comes from the fact that the function to be minimized at iteration t majorize the randomly chosen function $g_i(\cdot)$. More precisely, let M be a positive definite matrix such that $M \succeq \beta I$ for some $\beta > 0$, we define an auxiliary function $\Psi_\beta : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ by

$$\Psi_\beta(w_1, w_2) = \frac{1}{m} \sum_{i=1}^{m} \Psi_\beta^i(w_1, w_2), \tag{3.5}$$

where, for all $i = 1, 2, \ldots, m$, we have

$$\Psi_\beta^i(w_1, w_2) = \max_{\hat{y} \in Y} \{ g_{i,\hat{y}}(w_1) + \langle \nabla g_{i,\hat{y}}(w_1), w_2 - w_1 \rangle \} + \frac{\beta}{2} \|w_2 - w_1\|_M^2. \tag{3.6}$$

In the following result we prove that the function Ψ_β indeed majorize the objective function g and we also prove that critical points of this majorizing function are critical points of the original function g. For the case of $m = 1$ see similar results in [2, 5].

Proposition 3.1 (Properties of Ψ_β)**.** *Let M be a positive definite matrix such that $M \succeq \beta I$ for some. The following statements hold true:*

(i) $\Psi_\beta(w_1, w_1) = g(w_1)$ *for all* $w_1 \in \mathbb{R}^p$.

(ii) Suppose that $\beta \geq \sum_{i=1}^{m} \max_{\hat{y} \in Y} L_{i,\hat{y}}$. *Then, for all* $w_2 \in \mathbb{R}^p$, *we have*

$$\Psi_\beta(w_1, w_2) \geq g(w_2).$$

(iii) $\partial_{w_2} \Psi_\beta(w_1, w_1) \subseteq \partial g(w_1)$, *for all* $w_1 \in \mathbb{R}^p$.

Proof. The first item follows by a simple substitution of $w_2 = w_1$ in (3.5) and (3.6). The proof of the second item follows along the lines of the proof of Lemma 3.2. Now we will prove the last item. Using [23, Exercise 8.31], we have that the subdifferential of g_i (see (2.1)) is given by

$$\partial g_i(w_1) = \sum_{\hat{y} \in \widehat{Y}(w_1)} \lambda_{\hat{y}} \nabla g_{i,\hat{y}}(w_1), \tag{3.7}$$

where $\lambda \in \Delta^n := \left\{ \lambda \in \mathbb{R}_+^n : \sum_{j=1}^n \lambda_j = 1 \right\}$ (here $n = |\widehat{Y}(w_1)|$) and

$$\widehat{Y}(w_1) = \{ \hat{y} \in Y : g_{i,\hat{y}}(w_1) = g_i(w_1) \}.$$

The subdifferential of Ψ^i_β with respect to w_2, is given by

$$\partial_{w_2}\Psi^i_\beta(w_1,w_2)=\sum_{\hat{y}\in\widetilde{Y}(w_1,w_2)}\lambda_{\hat{y}}\nabla g_{i,\hat{y}}(w_1), \tag{3.8}$$

where

$$\widetilde{Y}(w_1,w_2)=\left\{\hat{y}\in Y:\ h_{i,\hat{y}}(w_1,w_2)=\max_{y\in Y}h_{i,y}(w_1,w_2)\right\}. \tag{3.9}$$

Setting $w_2=w_1$ in (3.9) yields

$$\widetilde{Y}(w_1,w_1)=\{\hat{y}\in Y:\ g_{i,\hat{y}}(w_1)=g_i(w_1)\}=\widehat{Y}(w_1).$$

This proves that $\partial_{w_2}\Psi^i_\beta(w_1,w_1)\subseteq\partial g_i(w_1)$ for all $i=1,2,\ldots,m$. By summing this inclusion over $1\leq i\leq m$ and using Lemma 3.1, the desired result follows. □

3.2 Convergence Analysis of SPLM

In this section we will present our main result concerning the convergence and rate of SPLM. We show that the sequence generated by SPLM almost surely (in short a.s.) converges to a critical point of the problem at hand.

To this end we use an inherent property that holds in our setting, defined in (2.1), namely that for every $i=1,2,\ldots,m$ there exists y for which $g_{i,y}(w)=0$. The property holds since the hinge-loss on the correct prediction is zero.

Our convergence and rate of convergence analysis of SPLM will be divided into two steps. We first show that the random sequences $g\left(w^0\right),g\left(w^1\right),\ldots$ and subgradients $u^0,u^1,\ldots$ that are generated by the SPLM algorithm form a supermartingale with respect to $\mathbb{F}_t:=\sigma\left(w^1,w^2,\ldots,w^t\right)$. Let $i=1,2,\ldots,m$. For simplicity we denote

$$H(w_1,w_2):=\frac{1}{m}\sum_{i=1}^m H_i(w_1,w_2)\quad\text{where}\quad H_i(w_1,w_2):=\max_{\hat{y}\in Y}h_{i,\hat{y}}(w_1,w_2), \tag{3.10}$$

and $h_{i,\hat{y}}(w_1,w_2)=g_{i,\hat{y}}(w_1)+\langle\nabla g_{i,\hat{y}}(w_1),w_2-w_1\rangle$. We denote by $\partial_2 H_i(w_1,w_2)$ the subdifferential of $H_i(\cdot,\cdot)$ with respect to the second variable.

Proposition 3.2. *Let $\left\{w^t\right\}_{t\in\mathbb{N}}$ be a sequence generated by SPLM and suppose that $M\succeq\beta I$, where $\beta=Lm/\gamma$, $\gamma<2$ and $L=\frac{1}{m}\sum_{i=1}^m\max_{\hat{y}\in Y}L_{i,\hat{y}}$. Define, for all $t\in\mathbb{N}$, the random variables $G_t=g\left(w^t\right)$ and*

$$U_t=c\sum_{i=1}^m\left\|u_i^t\right\|^2_{M^{-2}},\quad\textit{where}\quad u_i^t\in\partial_2 H_i\left(w^{t+1},w^t\right)\quad\textit{and}\quad c=\frac{\beta}{m}\left(\frac{1}{m}-\frac{L}{2\beta}\right). \tag{3.11}$$

Then, $\mathbb{E}\left[G_{t+1}|\,\mathbb{F}_t\right]+U_t\leq G_t$ for all $t\in\mathbb{N}$. In particular, $\{G_t\}_{t\in\mathbb{N}}$ a.s. converges to a non-negative random variable G_∞ and $\{U_t\}_{t\in\mathbb{N}}$ a.s. converges to 0.

Proof. Since $g_{i,\hat{y}}(w)\equiv 0$, $i=1,2,\ldots,m$, for some $\hat{y}\in Y$, it follows from (2.1) that g_i is non-negative function, and therefore g is also a non-negative function, implying the non-negativity of G_t, $t\in\mathbb{N}$.

Plugging $u = w^{t+1}$ and $v = w^t$ in Lemma 3.2 and taking expectation conditioned on $\mathbb{F}_t$ yields

$$\mathbb{E}\left[g\left(w^{t+1}\right)\middle|\,\mathbb{F}_t\right] \leq g\left(w^t\right) + \frac{1}{m}\sum_{i=1}^{m}\mathbb{E}\left[\max_{\hat{y}\in Y}\left\langle \nabla g_{i,\hat{y}}\left(w^t\right), w^{t+1}-w^t\right\rangle\middle|\,\mathbb{F}_t\right] + \frac{L}{2}\mathbb{E}\left[\left\|w^{t+1}-w^t\right\|^2\middle|\,\mathbb{F}_t\right]. \tag{3.12}$$

Using the optimality condition of w^{t+1} (see (3.1)), there exists $u_i^t \in \partial H_i\left(w^t, w^{t+1}\right)$ such that

$$u_i^t + M\left(w^{t+1}-w^t\right) = \mathbf{0}. \tag{3.13}$$

From the definition of $H_i\left(w, w^t\right)$, $i = 1, 2, \ldots, m$, as the maximum of linear functions it follows that

$$\partial_2 H_i\left(w^{t+1}, w^t\right) = \operatorname{conv}\left(\left\{\nabla g_{i,\hat{y}}\left(w^t\right)\right\}_{\hat{y}\in\bar{Y}}\right), \tag{3.14}$$

where $\bar{Y} := \left\{\bar{y}\in Y:\ H_i\left(w^{t+1}, w^t\right) = h_{i,\hat{y}}\left(w^{t+1}, w^t\right)\right\}$. This means that

$$\mathbb{E}\left[{u_i}^t\middle|\,\mathbb{F}_t\right] = (1/m)\sum_{l=1}^{m} u_l^t.$$

Combining (3.12) and (3.13) yields

$$\begin{aligned}\mathbb{E}\left[g\left(w^{t+1}\right)\middle|\,\mathbb{F}_t\right] &\leq g\left(w^t\right) - \frac{1}{m^2}\sum_{i=1}^{m}\sum_{l=1}^{m}\max_{\hat{y}\in Y}\left\langle \nabla g_{i,\hat{y}}\left(w^t\right), M^{-1}u_l^t\right\rangle + \frac{L}{2m}\sum_{i=1}^{m}\left\|u_i^t\right\|_{M^{-2}}^2 \\ &\leq g\left(w^t\right) - \frac{1}{m^2}\sum_{i=1}^{m}\max_{\hat{y}\in Y}\left\langle \nabla g_{i,\hat{y}}\left(w^t\right), M^{-1}u_i^t\right\rangle + \frac{L}{2m}\sum_{i=1}^{m}\left\|u_i^t\right\|_{M^{-2}}^2, \end{aligned} \tag{3.15}$$

where the second inequality is due to the fact that $\nabla g_{i,\hat{y}}(w^t) = 0$ for some $\hat{y}\in Y$, $i = 1, 2, \ldots, m$, and thus

$$\max_{\hat{y}\in Y}\left\langle \nabla g_{i,\hat{y}}\left(w^t\right), M^{-1}u_l^t\right\rangle \geq 0, \quad 1\leq i, l\leq m.$$

On the other hand, from (3.14) we have that

$$u_i^t = \sum_{\hat{y}\in\bar{Y}}\lambda_{\hat{y}}\nabla g_{i,\hat{y}}\left(w^t\right),$$

where $\lambda\in\Delta^n$. In addition, for all $i = 1, 2, \ldots, m$, we have that

$$\begin{aligned}\left\|u_i^t\right\|_{M^{-2}}^2 = \left\|M^{-1}u_i^t\right\|^2 &\leq \left\|M^{-1/2}\right\|^2\left\|M^{-1/2}u_i^t\right\|^2 \\ &\leq \frac{1}{\beta}\sum_{\hat{y}\in\bar{Y}}\lambda_{\hat{y}}\left\langle M^{-1}u_i^t, \nabla g_{i,\hat{y}}\left(w^t\right)\right\rangle \\ &\leq \frac{1}{\beta}\max_{\hat{y}\in\bar{Y}}\left\langle M^{-1}u_i^t, \nabla g_{i,\hat{y}}\left(w^t\right)\right\rangle \\ &\leq \frac{1}{\beta}\max_{\hat{y}\in Y}\left\langle M^{-1}u_i^t, \nabla g_{i,\hat{y}}\left(w^t\right)\right\rangle, \end{aligned}$$

where the first inequality is due to the fact that $M\succeq\beta I$, the second inequality follows from the fact that $\lambda\in\Delta^n$ and the last inequality follows from the fact that $\bar{Y}\subset Y$. Using this fact in (3.15)

yields that

$$\mathbb{E}\left[g\left(w^{t+1}\right)\middle|\mathbb{F}_t\right] \leq g\left(w^t\right) - \frac{\beta}{m^2}\sum_{i=1}^{m}\left\|u_i^t\right\|_{M^{-2}}^2 + \frac{L}{2m}\sum_{i=1}^{m}\left\|u_i^t\right\|_{M^{-2}}^2$$
$$= g\left(w^t\right) - \frac{\beta}{m}\left(\frac{1}{m} - \frac{L}{2\beta}\right)\sum_{i=1}^{m}\left\|u_i^t\right\|_{M^{-2}}^2.$$

From the definition of U_t (see (3.11)) we obtain that

$$\mathbb{E}\left[G_{t+1}\middle|\mathbb{F}_t\right] + U_t \leq G_t, \quad \forall\, t \in \mathbb{N}. \tag{3.16}$$

Since we assumed that $\gamma < 2$, it is clear that $\beta \geq Lm/2$ and we have that $U_t \geq 0$. Using the supermartingale convergence theorem, combined with (3.16) and the fact that G_t and U_t, $t \in \mathbb{N}$, are non-negative random variables, we obtain that $\{G_t\}_{t\in\mathbb{N}}$ converges a.s., as $t \to \infty$, to a certain non-negative random variable G_∞ and that $\{U_t\}_{t\in\mathbb{N}}$ converges a.s. to 0, as required. □

The above result proves that objective function values $G_t = g\left(w^t\right)$ that are generated by the SPLM algorithm is almost surely converges to a certain limit point G_∞. It also proves that the (sub)gradients of $H\left(w^{t+1}, w^t\right)$, $t \in \mathbb{N}$, almost surely converges to zero, namely, a limiting point of the sequence of (sub)gradients is a critical point of H and therefore from Proposition 3.1 is a critical point of the original objective function g.

Now we are ready to prove our main result about the rate of convergence of SPLM.

Theorem 3.1. *Let $\left\{w^t\right\}_{t\in\mathbb{N}}$ be a sequence generated by SPLM. Assume that g is bounded from below on $\mathbb{R}^d$ by some $\bar{g}$. Let $u^t = \frac{1}{m}\sum_{i=1}^{m} u_i^t$ where $u_i^t \in \partial H_i\left(w^{t+1}, w^t\right)$, $i = 1, 2, \ldots, m$, and suppose that $M \succeq \beta I$, where $\beta = Lm/\gamma$, $\gamma < 2$ and $L = \frac{1}{m}\sum_{i=1}^{m}\max_{\hat{y}\in Y} L_{i,\hat{y}}$. Then, we have that*

$$\min_{t=0,1,\ldots,T-1}\mathbb{E}\left[\left\|u^t\right\|_{M^{-2}}\right] \leq \varepsilon, \tag{3.17}$$

after $T \geq \frac{L(2-\gamma)}{2\gamma\varepsilon^2}\left(g\left(w^0\right) - \bar{g}\right)$ iterations.

Proof. Suppose now that $\beta = Lm/\gamma$ where $\gamma < 2$. Thus, in this case, we have that U_t (as defined in (3.11)) is positive for all $t \in \mathbb{N}$. Plugging β into U_t yields

$$U_t = c\sum_{i=1}^{m}\left\|u_i^t\right\|_{M^{-2}}^2, \tag{3.18}$$

where

$$c = \frac{L(2-\gamma)}{2\gamma m}. \tag{3.19}$$

Since we proved in Proposition 3.2 that $\{Y_t\}_{t\in\mathbb{N}}$ converges a.s. to 0, it follows that $\left\{u_i^t\right\}_{t\in\mathbb{N}}$ also converges a.s. to $\mathbf{0}$ for all $i = 1, 2, \ldots, m$. Taking expectation of (3.11) yields that

$$\mathbb{E}\left[U_t\right] \leq \mathbb{E}\left[G_t\right] - \mathbb{E}\left[G_{t+1}\right]. \tag{3.20}$$

Summing (3.20) over $t = 0, 1, \ldots, T-1$ and dividing by T yields

$$\min_{t=0,1,\ldots,T-1}\mathbb{E}\left[U_t\right] \leq \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[U_t\right] \leq \frac{1}{T}\left(\mathbb{E}\left[G_0\right] - \mathbb{E}\left[G_T\right]\right) \leq \frac{1}{T}\left(g\left(w^0\right) - \bar{g}\right), \tag{3.21}$$

where the first inequality holds due to the fact that the minimum is less than the average. From (3.18) and the convexity of $\|\cdot\|_{M^{-2}}^2$ we obtain

$$\left\|u^t\right\|_{M^{-2}}^2 = \left\|\frac{1}{m}\sum_{i=1}^m u_i^t\right\|_{M^{-2}}^2 \leq \frac{1}{m}\sum_{i=1}^m \left\|u_i^t\right\|_{M^{-2}}^2 = \frac{1}{mc}U_t. \tag{3.22}$$

Combining (3.22) together with (3.21), using Jensen's inequality and plugging c (see (3.19)) yields

$$\begin{aligned}\min_{t=0,1,\ldots,T-1}\mathbb{E}\left[\left\|u^t\right\|_{M^{-2}}\right]_{M^{-2}}^2 &\leq \min_{t=0,1,\ldots,T-1}\mathbb{E}\left[\left\|u^t\right\|_{M^{-2}}^2\right] \\ &\leq \frac{1}{mcT}\left(g\left(w^0\right)-\bar{g}\right) \\ &= \frac{2\gamma}{(2-\gamma)LT}\left(g\left(w^0\right)-\bar{g}\right),\end{aligned}$$

which proves the desired sub-linear rate of convergence result. □

An immediate consequence of Theorem 3.1 is the non-asymptotic rate of convergence of the sequence $\left\{\left\|w^{t+1}-w^t\right\|\right\}_{t\in\mathbb{N}}$. Indeed, from (3.21) it follows that

$$\min_{t=0,1,\ldots,T-1}\mathbb{E}\left[\left\|w^{t+1}-w^t\right\|\right] \leq \sqrt{\frac{2m^2\gamma}{(2-\gamma)\,LT}\left(g\left(w^0\right)-\bar{g}\right)}.$$

To conclude, in this work we presented a novel algorithm for finding critical points of non-convex and non-smooth function, whose non-smoothness is given by the maximum of smooth and non-convex functions. The proposed algorithm combines stochastic gradient descent with the Proximal Linear Method, and minimizes at each iteration a majorizing function, whose critical points included in the set of critical points of the original objective function, with a rate of $O(m^2/\varepsilon^2)$. We demonstrate below the effectiveness of our approach on learning VGG networks.

It should be noted that the studied model can not be tackled by Stochastic Gradient Descent (SGD) and its variants since they all require a smooth component in their objective function. Therefore, to the best of our knowledge, the only algorithm that can be used in this setting is the Stochastic Sub-Gradient Descent (SSGD). However, the SSGD has no provable rate of convergence in the non-convex setting. On the other hand, we have shown that the proposed SPLM algorithm converges, namely the norm of the subgradient vanishes, at rate of $O(m^2/\varepsilon^2)$. As our experiments below demonstrate, also in practice the proposed SPLM algorithm is significantly better than SSGD.

4 Experimental Evaluation

4.1 Solving the Inner Optimization Problem of SPLM

Fix $t \geq 1$. Each iteration of SPLM requires solving an optimization problem (see (3.1)) that can be written as follows

$$\min_{w\in\mathbb{R}^p}\left\{\max_{\hat{y}\in Y}\left\{\left\langle a_{\hat{y}}, w\right\rangle + b_{\hat{y}}\right\} + \frac{\beta}{2}\left\|w-w^t\right\|^2\right\}, \tag{4.1}$$

where $a_{\hat{y}}^t = \nabla g_{i,\hat{y}}\left(w^t\right)$ and $b_{\hat{y}}^t = g_{i,\hat{y}}\left(w^t\right) - \left\langle \nabla g_{i,\hat{y}}\left(w^t\right), w^t\right\rangle$. Let $A \in \mathbb{R}^{|Y|\times p}$ be a matrix whose columns are the vectors $a_{\hat{y}}^t$ for $\hat{y} \in Y$.

As mentioned in Section 3, at each iteration we apply a predefined number (denoted by K) of iteration of the FDPG method of Beck and Teboulle [3] to solve the minimization problem (4.1). The explicit steps of the FDPG method in our case are as follows.

FDPG for minimization step in SPLM

0. Input: fix $L \geq \frac{\|A\|^2}{\beta}$ and $K \in \mathbb{N}$.
1. Initialization: start with any $z_1 = y_0 \in -\frac{1}{L}\Delta_n$, $t_1 = 1$.
2. For each $k = 1, 2, \ldots, K$ compute

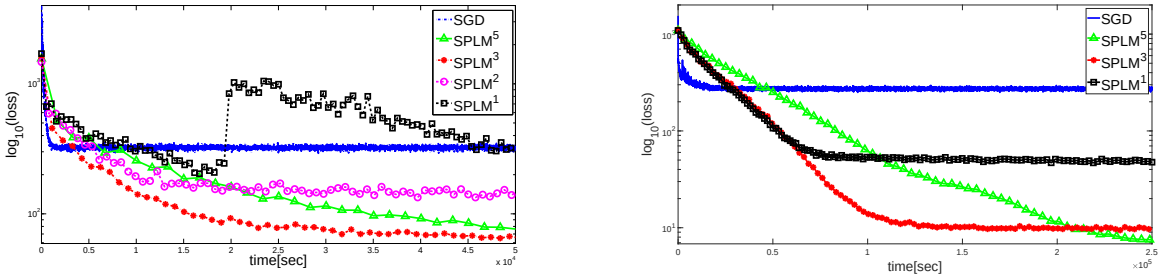
$$\begin{aligned} \bar{z}_k &= w^t + \frac{1}{\beta} A^T z_k \\ y_k &= -\frac{1}{L} P_{\Delta_n}\left(\frac{1}{L}\left(A\bar{z}_k + b_{\hat{y}}^t - L z_k\right)\right) \\ t_{k+1} &= \frac{1+\sqrt{1+4t_k^2}}{2} \\ z_{k+1} &= y_k + \frac{t_k - 1}{t_{k+1}}\left(y_k - y_{k+1}\right) \end{aligned}$$

We applied the SPLM algorithm to solve the optimization problem during the training of a neural network. We compared SPLM to SGD using a smoothed version of the VGG network, i.e., we used the transfer function $f(t) = \log(1 + \exp(t))$, and replaced the max-pooling components with the mean-pooling. The total number of parameters in the network denoted by p is about 15 million. In our experiments we used the CIFAR10 dataset.

Our first experiments designed to check the influence of the inner iterations of the FDPG method on the overall performances, in terms of the achieved objective function values, of the SPLM algorithm. To this end we have tried four configurations of number of inner iterations: 1, 2, 3 and 5. Figure 1 depicts the loss of the SGD and the SPLM^k algorithms versus time, where k stands for the number of inner FDPG iterations used to solve the optimization problem at each iteration of the SPLM.

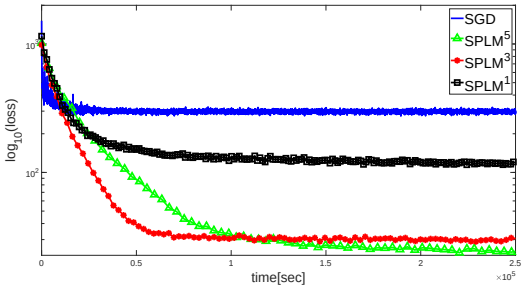
As can be seen from this experiment, 3 inner iterations is a good balance between the time we spent on solving the inner sub-problems and the overall performance of the algorithm. Therefore, in our next experiment we fixed the number of inner iterations to be 3.

Next, we also compared the performance of the SPLM algorithm to the ADAM algorithm (see [14]), which is a very popular algorithm for training deep neural networks. The figure below depicts the loss of SGD, SPLM and ADAM for the smoothed network with again 15 million parameters, and provides a very promising evidence for the superiority of SPLM over the two other methods.

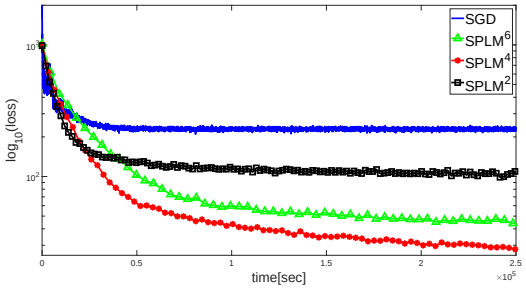


(a) Batch size $= 1$, $\beta = 0.0001$

(b) Batch size $= 5$, $\beta = 0.001$



(c) Batch size $= 10$, $\beta = 0.01$



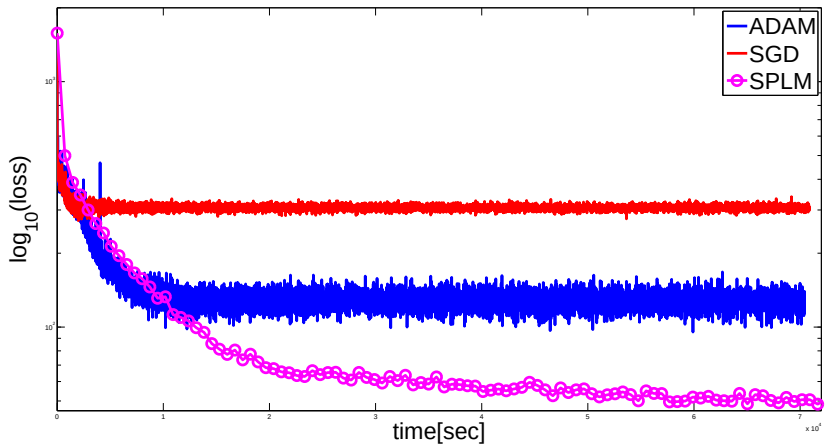
(d) Batch size $= 10$, $\beta = 0.02$

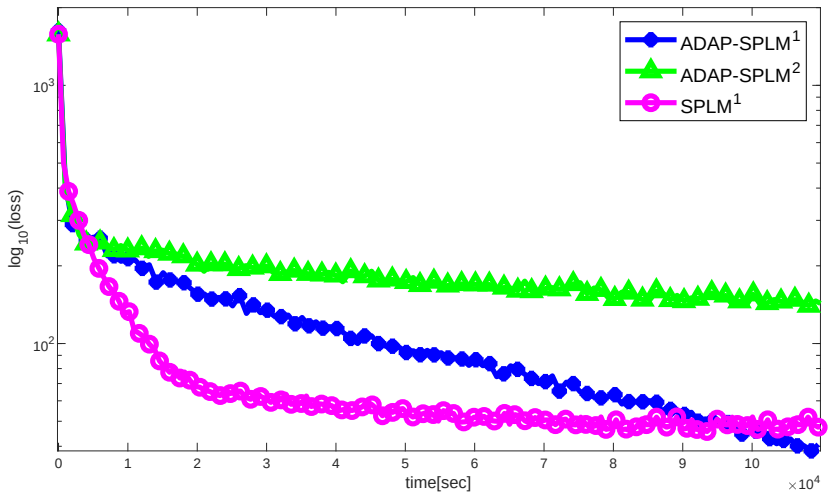
Figure 1: Comparing SPLM to SGD

In another experiment, we have tested the effect of changing the weight matrices M_t, $t \in \mathbb{N}$. In this case, we have exactly used the idea of [11], where they suggest to incorporate the previously computed gradients. Here, in each experiments, we used variety if step-sizes and also checked different numbers of inner FDPG iterations (denoted with n superscript number). First, we compared the non-adaptive SPLM with the matrix $M_t = \beta I$ (where $\beta = 0.001$) to two versions of the adaptive SPLM (denoted below by ADAP-SPLM), for which the weight matrices satisfy $M_t \succeq \beta_1 I$ and $M_t \succeq \beta_2 I$ (where $\beta_1 = 0.001$ and $\beta_2 = 0.005$).

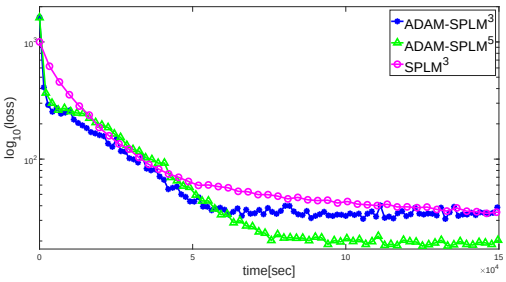
This experiment provide an evidence that the adaptive version of SPLM can perform better for suitable parameters of β and even for smaller number of inner iterations.

Lastly, we implemented the ADAM version of SPLM (denoted below by ADAM-SPLM), mean-

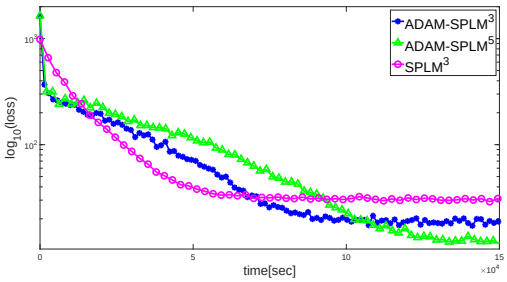




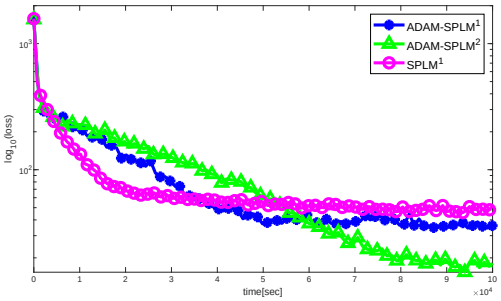
ing the weights of the initial gradients used to compute the matrix M_t decay over the iterates t. Again, we have checked different step-size and number of inner iterations.



(a) SPLM with $\beta = 0.01$



(b) SPLM with $\beta = 0.02$



(c) SPLM with $\beta = 0.001$

We see clearly that this adaptive version of the SPLM method outperforms the non-adaptive version in all cases.

References

[1] N. Agarwal, Z. Allen-Zhu, B. Bullins, E. Hazan and T. Ma. Finding approximate local minima for nonconvex optimization in linear time. *Symposium on Theory of Computing*, 2017.

[2] A. Auslender. An extended sequential quadratically constrained quadratic programming algorithm for nonlinear, semidefinite, and second-order cone programming. *J. Optim. Theory Appl.*, 156(2):183–212, 2013.

[3] A. Beck and M. Teboulle. A fast dual proximal gradient algorithm for convex minimization and applications. *Oper. Res. Lett.*, 42(1):1–6, 2014.

[4] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Englewood Cliffs, NJ: Prentice hall, 1989.

[5] J. Bolte and E. Pauwels. Majorization-minimization procedures and convergence of SQP methods for semi-algebraic and tame programs. *Math. Oper. Res.*, 41(2):442–465, 2016.

[6] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

[7] Y. Carmon, J. C. Duchi, O. Hinder and A. Sidford. Convex until proven guilty: Dimension-free acceleration of gradient descent on non-convex functions. *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 654–663, 2017.

[8] Y. Carmon, , J. C. Duchi, O. Hinder and A. Sidford. Accelerated methods for nonconvex optimization. *SIAM Journal on Optimization*, 28(2):1751–1772, 2018.

[9] D. Davis, B. Edmunds and M. Udell. The sound of apalm clapping: Faster nonsmooth nonconvex optimization with stochastic asynchronous palm. *Advances in Neural Information Processing Systems*, 226–234, 2016.

[10] A. Defazio, F. Bach and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 1646–1654, 2014.

[11] Y. Duchi, E. Hazan and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, 2011.

[12] R. Ge, F. Huang, C. Jin and Y. Yuan. Escaping from saddle points-online stochastic gradient for tensor decomposition. *Conference on Learning Theory*, 797–842, 2015.

[13] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 315–323, 2013.

[14] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] H. Lee and H. Kwon. Going deeper with contextual CNN for hyperspectral image classification. *IEEE T. Image Process.*, 26(10):4843–4855, 2017.

[16] J. D. Lee, M. Simchowitz, M. I. Jordan and B. Recht. Gradient descent only converges to minimizers. *Conference on Learning Theory*, 1246–1257, 2016.

[17] A. S. Lewis, and S. J. Wright. A proximal method for composite minimization. *Math. Program.*, 158(1-2):501–546, 2016.

[18] S. Liu, B. Kailkhura, P. Y. Chen, P. Ting, S. Chang and L. Amini. Zeroth-order stochastic variance reduction for nonconvex optimization. *Advances in Neural Information Processing Systems*, 3727–3737, 2018.

[19] E. Pauwels. The value function approach to convergence analysis in composite optimization. *Oper. Res. Lett.*, 44(6):790–795, 2016.

[20] S. J. Reddi, A. Hefny, S. Sra, B. Poczos and A. Smola. Stochastic variance reduction for nonconvex optimization. *International Conference on Machine Learning*, 314–323, 2016.

[21] H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Stat.*, 400–407, 1951.

[22] R. T. Rockafellar. *Convex Analysis*. Princeton Mathematical Series, No. 28. Princeton University Press, Princeton, N.J., 1970.

[23] R. T. Rockafellar and R. J.-B. Wets. *Variational Analysis*, volume 317. Springer-Verlag, Berlin, 1998.

[24] M. Schmidt and N. Le Roux and F. Bach. Minimizing finite sums with the stochastic average gradient. *Math. Program.*, 162(1-2):1–30, 2017.

[25] S. Shalev-Shwartz, Y. Singer, N. Srebro and A. Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Math. Program.*, 127(1):83–112, 2011.

[26] M. Staib, S. J. Reddi, S. Kale, S. Kumar and S. Sra. Escaping saddle points with adaptive gradient methods. *International Conference on Machine Learning*, 5956–5965, 2019.

[27] M. Teboulle. A simplified view of first order methods for optimization. *Math. Program.*, 170:67–96, 2018.

[28] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang and W. Xu. CNN-RNN: A unified framework for multi-label image classification. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5–16, 2016.