OPT-AMSGrad: An Optimistic Acceleration of AMSGrad for Nonconvex Optimization

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

We consider a new variant of AMSGrad. AMSGrad [23] is a popular adaptive gradient based optimization algorithm that is widely used in training deep neural networks. The new variant assumes that mini-batch gradients in consecutive iterations have some underlying structure, which makes the gradients sequentially predictable. By exploiting the predictability and ideas from Optimistic Online Learning, the proposed algorithm can accelerate the convergence and increase sample efficiency. In the nonconvex case, we establish a $\mathcal{O}\left(\sqrt{d/T} + d/T\right)$ non-asymptotic bound independent of the initialization of the method and in the convex case, we show that our method enjoys a tighter non-asymptotic regret bound under some conditions. We illustrate the practical speedup on several deep learning models through numerical experiments.

1 Introduction

Deep learning models have been successful in several applications, from robotics (e.g. [17]), computer vision (e.g [14, 11]), reinforcement learning (e.g. [19]), to natural language processing (e.g. [12]). With the sheer size of modern data sets and the dimension of neural networks, speeding up training is of utmost importance. To do so, several algorithms have been proposed in recent years, such as AMSGrad [23], ADAM [15], RMSPROP [27], ADADELTA [30], and NADAM [7].

All the prevalent algorithms for training deep networks mentioned above combine two ideas: the idea of adaptivity from AdaGrad [8, 18] and the idea of momentum from Nesterov's Method [20] or Heavy ball method [21]. AdaGrad is an online learning algorithm that works well compared to the standard online gradient descent when the gradient is sparse. Its update has a notable feature: it leverages an anisotropic learning rate depending on the magnitude of gradient in each dimension which helps in exploiting the geometry of data. On the other hand, Nesterov's Method or Heavy ball Method [21] is an accelerated optimization algorithm whose update not only depends on the current iterate and current gradient but also depends on the past gradients (i.e. momentum). State-of-the-art algorithms like AMSGrad [23] and ADAM [15] leverage these ideas to accelerate the training process of highly nonconvex objective functions such as deep neural networks losses.

In this paper, we propose an algorithm that goes further than the hybrid of the adaptivity and momentum approach. Our algorithm is inspired by Optimistic Online learning [4, 22, 26, 1], which assumes that a good *guess* of the loss function in each round of online learning is available, and plays an action by exploiting the guess. By exploiting the guess, algorithms in Optimistic Online learning enjoy smaller regret than the ones without exploiting the guess. We combine the Optimistic Online learning idea with the adaptivity and the momentum ideas to design a new algorithm — OPT-AMSGrad. To the best of our knowledge, this is the first work exploring towards this direction. The proposed algorithm not only adapts to the informative dimensions, exhibits momentum, but also exploits a good guess of the next gradient to facilitate acceleration. Besides the global analysis of OPT-AMSGrad, we also conduct experiments and show that the proposed algorithm not only accelerates convergence of loss function, but also leads to better generalization performance in some cases.

2 Preliminaries

We begin by providing some background on both online learning and adaptive methods.

Notations: We follow the notations in related adaptive optimization papers [15, 23]. For any vector u, $v \in \mathbb{R}^d$, u/v represents element-wise division, u^2 represents element-wise square, $\sqrt{u}$ represents element-wise square-root. We denote $g_{1:T}[i]$ as the sum of the i_{th} element of T vectors $g_1, g_2, \dots, g_T \in \mathbb{R}^d$.

2.1 Optimistic Online learning

The standard setup of Online learning is that, in each round t, an online learner selects an action $w_t \in \Theta \subseteq \mathbb{R}^d$, then the learner observes $\ell_t(\cdot)$ and suffers loss $\ell_t(w_t)$ after the action is committed. The goal of the learner is to minimize the regret,

$$\text{Regret}_T(\{w_t\}) := \sum_{t=1}^{T} \ell_t(w_t) - \sum_{t=1}^{T} \ell_t(w^*) ,$$

which is the cumulative loss of the learner minus the cumulative loss of some benchmark $w^* \in \Theta$. The idea of Optimistic Online learning (e.g. [4, 22, 26, 1]) is as follows. In each round t, the learner exploits a good guess $m_t(\cdot)$ of the gradient $\nabla \ell_t(\cdot)$ of the loss function to choose an action w_t. [1] Consider the Follow-the-Regularized-Leader (FTRL, [13]) online learning algorithm which update reads

$$w_t = \arg\min_{w \in \Theta} \langle w, L_{t-1} \rangle + \tfrac{1}{\eta} R(w) , \tag{1}$$

where η is a parameter, $R(\cdot)$ is a 1-strongly convex function with respect to a norm ($\|\cdot\|$) on the constraint set Θ, and $L_{t-1} := \sum_{s=1}^{t-1} g_s$ is the cumulative sum of gradient vectors of the loss functions up to $t-1$. It has been shown that FTRL has regret at most $O(\sqrt{\sum_{t=1}^{T} \|g_t\|_*})$. The update of its optimistic variant, noted Optimistic-FTRL and developed in [26] reads

$$w_t = \arg\min_{w \in \Theta} \langle w, L_{t-1} + m_t \rangle + \tfrac{1}{\eta} R(w) , \tag{2}$$

where m_t is the learner's guess of the gradient vector $g_t := \nabla \ell_t(w_t)$. Under the assumption that loss functions are convex, the regret of Optimistic-FTRL is at most $O(\sqrt{\sum_{t=1}^{T} \|g_t - m_t\|_*})$, which can be much smaller than the regret of FTRL if m_t is close to g_t. Consequently, Optimistic-FTRL can achieve better performance than FTRL. On the other hand, if m_t is far from g_t, then the regret of Optimistic-FTRL is only a constant factor worse than that of its counterpart FTRL.

We emphasize in Section 3 the importance of leveraging a good guess m_t for updating w_t in order to get a fast convergence rate (or equivalently, small regret). We will have a similar argument when we compare OPT-AMSGrad and AMSGrad.

2.2 Adaptive optimization methods

Recently, adaptive optimization has been popular in various deep learning applications due to their superior empirical performance. Adam [15] is a very popular adaptive algorithm for training deep nets. It combines the momentum idea [21] with the idea of AdaGrad [8], which has different learning rates for different dimensions, adaptive to the learning process. More specifically, the learning rate of AdaGrad in iteration t for a dimension j is proportional to the inverse of $\sqrt{\Sigma_{s=1}^{t} g_s[j]^2}$, where $g_s[j]$ is the j-th element of the gradient vector g_s at time s.

[1] Imagine that if the learner would had been known $\nabla \ell_t(\cdot)$ (*i.e.,* exact guess) before committing its action, then it would exploit the knowledge to determine its action and consequently minimizes the regret.

This adaptive learning rate helps accelerating the convergence when the gradient vector is sparse [8] but, when applying AdaGrad to train deep networks, it is observed that the learning rate might decay too fast [15]. Therefore, [15] proposes Adam that uses a moving average of gradients divided by the square root of the second moment of the moving average (element-wise fashion), for updating the model parameter w. A variant, called AMSGrad and detailed in Algorithm 1, has been developed in [23] to fix Adam failures at some online convex optimization problems. The difference between Adam and AMSGrad lies in line 7 of Algorithm 1. Adam does not have the max operation on line 7 (i.e. $\hat{v}_t = v_t$ for Adam) while [23] adds the operation to guarantee a non-increasing learning rate, $\frac{\eta_t}{\sqrt{\hat{v}_t}}$, which helps for the convergence (i.e. average regret $\frac{\text{Regret}_T}{T} \to 0$). For the hyper-parameters of AMSGrad, it is suggested in [23] that $\beta_1 = 0.9$ and $\beta_2 = 0.99$.

Algorithm 1 AMSGrad [23]

```
1: Required: parameter $\beta_1$, $\beta_2$, and $\eta_t$.
2: Init: $w_1 \in \Theta \subseteq \mathbb{R}^d$ and $v_0 = \epsilon 1 \in \mathbb{R}^d$.
3: for $t = 1$ to $T$ do
4:     Get mini-batch stochastic gradient $g_t$ at $w_t$.
5:     $\theta_t = \beta_1 \theta_{t-1} + (1 - \beta_1) g_t$.
6:     $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$.
7:     $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$.
8:     $w_{t+1} = w_t - \eta_t \frac{\theta_t}{\sqrt{\hat{v}_t}}$. (element-wise division)
9: end for
```

3 OPT-AMSGrad Algorithm

We formulate in this section the proposed optimistic acceleration of AMSGrad, noted OPT-AMSGrad, and detailed in Algorithm 2.

Algorithm 2 OPT-AMSGrad

```
1: Required: parameter $\beta_1$, $\beta_2$, $\epsilon$, and $\eta_t$.
2: Init: $w_1 = w_{-1/2} \in \Theta \subseteq \mathbb{R}^d$ and $v_0 = \epsilon 1 \in \mathbb{R}^d$.
3: for $t = 1$ to $T$ do
4:     Get mini-batch stochastic gradient $g_t$ at $w_t$.
5:     $\theta_t = \beta_1 \theta_{t-1} + (1 - \beta_1) g_t$.
6:     $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$.
7:     $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$.
8:     $w_{t+\frac{1}{2}} = w_t - \eta_t \frac{\theta_t}{\sqrt{\hat{v}_t}}$.
9:     $w_{t+1} = w_{t+\frac{1}{2}} - \eta_t \frac{h_{t+1}}{\sqrt{\hat{v}_t}}$,
       where $h_{t+1} := \beta_1 \theta_{t-1} + (1 - \beta_1) m_{t+1}$ and $m_{t+1}$ is the guess of $g_{t+1}$.
10: end for
```

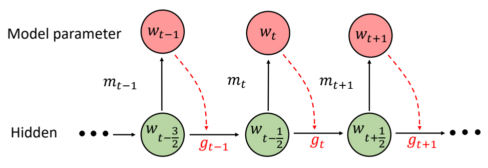


Figure 1: OPT-AMSGRAD.

It combines the idea of adaptive optimization with optimistic learning. At each iteration, the learner computes a gradient vector $g_t := \nabla \ell_t(w_t)$ at w_t (line 4), then it maintains an exponential moving average of $\theta_t \in \mathbb{R}^d$ (line 5) and $v_t \in \mathbb{R}^d$ (line 6), which is followed by the max operation to get $\hat{v}_t \in \mathbb{R}^d$ (line 7). The learner also updates an auxiliary variable $w_{t+\frac{1}{2}} \in \Theta$ (line 8). It uses the auxiliary variable (hidden model) to update and commit w_{t+1} (line 9), which exploits the guess m_{t+1} of g_{t+1} to get w_{t+1}. We summarize the interplay between the auxiliary structure $w_{t+\frac{1}{2}}$ and the model parameter w_{t+1} of AMSGrad in Figure 1.

The proposed OPT-AMSGrad inherits three properties:

- Adaptive learning rate of each dimension as AdaGrad [8]. (line 6, line 8 and line 9)
- Exponential moving average of the past gradients as Nesterov's method [20] and the Heavy-Ball method [21]. (line 5)
- Optimistic update that exploits a good guess of the next gradient vector as optimistic online learning algorithms [4, 22, 26]. (line 9)

The first property helps for acceleration when the gradient has a sparse structure. The second one is from the well-recognized idea of momentum which can also help for acceleration. The last one, perhaps less known outside the Online learning community, can actually lead to acceleration when the prediction of the next gradient is good. This property will be elaborated in the following subsection in which we provide the theoretical analysis of OPT-AMSGrad. Observe that the proposed algorithm does not reduce to AMSGrad when $m_t = 0$.

4 Global Convergence of OPT-AMSGrad

4.1 Convex Analysis: Finite-Time Regret Analysis

4.2 Nonconvex Analysis: Finite-Time Upper Bound

In this section, we discuss the offline and stochastic non-convex optimization properties of our online framework. In the stochastic optimization literature, the problem we are tackling reads as follows:

$$\min_{w \in \Theta} f(w) := \mathbb{E}[f(w, \xi)] \ , \tag{3}$$

where ξ is some random noise and only noisy versions of the objective function are accessible in this work. The objective function $f(w)$ is (potentially) nonconvex and has Lipschitz gradients.

Set the terminating iteration number, $T \in \{0, \ldots, T_{\mathsf{max}} - 1\}$, as a discrete r.v. with:

$$P(T = \ell) = \frac{\eta_\ell}{\sum_{j=0}^{T_{\mathsf{max}}-1} \eta_j} \ , \tag{4}$$

where T_{max} is the maximum number of iteration. The random termination number (4) is inspired by [10] which enables one to show non-asymptotic convergence to stationary point for non-convex optimization.

We make the following mild assumptions necessary to our analysis:

H1. *The loss function $f(w)$ is nonconvex* w.r.t. *the parameter w.*

H2. *For any $t > 0$, the estimated weight w_t stays within a ℓ_∞−ball. There exists a constant $W > 0$ such that:*

$$\|w_t\| \leq W \quad \textit{almost surely} \tag{5}$$

H3. *The function $f(w)$ is L-smooth (has L-Lipschitz gradients) w.r.t. the parameter w. There exist some constant $L > 0$ such that for $(w, \vartheta) \in \Theta^2$:*

$$f(w) - f(\vartheta) - \nabla f(\vartheta)^\top (w - \vartheta) \leq \frac{L}{2} \|w - \vartheta\|^2 \ . \tag{6}$$

We assume that the optimistic guess m_t at iteration k and the true gradient g_t are correlated:

H4. *There exists a constant $a \in \mathbb{R}$ such that for any $t > 0$:*

$$\langle m_t \,|\, g_t \rangle \leq a \|g_t\|^2$$

Classically in nonconvex optimization, see [10], we make an assumption on the magnitude of the gradient:

H5. *There exists a constant $\mathsf{M} > 0$ such that*

$$\|\nabla f(w, \xi)\| < \mathsf{M} \quad \textit{for any } w \textit{ and } \xi$$

We begin with some auxiliary Lemmas important for the analysis. The first one ensures bounded norms of various quantities of interests (resulting from the classical stochastic gradient boundedness assumption):

Lemma 1. *Assume assumption H 5, then the quantities defined in Algorithm* **??** *satisfy for any $w \in \Theta$ and $t > 0$:*

$$\|\nabla f(w_t)\| < \mathsf{M}, \quad \|\theta_t\| < \mathsf{M}, \quad \|\hat{v}_t\| < \mathsf{M}^2 \ .$$

See Proof in Appendix A.1.

Then, following [29] and their study of the SGD with Momentum (not AMSGrad but simple momentum) we denote for any $t > 0$:

$$\overline{w}_t = w_t + \frac{\beta_1}{1-\beta_1}(w_t - w_{t-1}) = \frac{1}{1-\beta_1} w_t - \frac{\beta_1}{1-\beta_1} w_{t-1} \, , \tag{7}$$

and derive an important Lemma:

Lemma 2. *Assume a strictly positive and non increasing sequence of stepsizes* $\{\eta_t\}_{t>0}$, $\beta \in [0,1]$, *then the following holds:*

$$\overline{w}_{t+1} - \overline{w}_t \leq \frac{\beta_1}{1-\beta_1}\tilde{\theta}_{t-1}\left[\eta_{t-1}\hat{v}_{t-1}^{-1/2} - \eta_t \hat{v}_t^{-1/2}\right] - \eta_t \hat{v}_t^{-1/2}\tilde{g}_t \, , \tag{8}$$

where $\tilde{\theta}_t = \theta_t + \beta_1\theta_{t-1}$ *and* $\tilde{g}_t = g_t - \beta_1 m_t + \beta_1 g_{t-1} + m_{t+1}$.

See Proof in Appendix A.2

Lemma 3. *Assume H 5, a strictly positive and a sequence of constant stepsizes* $\{\eta_t\}_{t>0}$, $\beta \in [0,1]$, *then the following holds:*

$$\sum_{k=1}^{T_{\textsf{max}}} \eta_t^2 \mathbb{E}\left[\left\|\hat{v}_t^{-1/2}\theta_t\right\|_2^2\right] \leq \frac{\eta^2 d T_{\textsf{max}}(1-\beta_1)}{(1-\beta_2)(1-\gamma)} \tag{9}$$

See Proof in Appendix A.3.

We now formulate the main result of our paper giving a finite-time upper bound of the quantity $\mathbb{E}\left[\|\nabla f(w_T)\|^2\right]$ where T is a random termination number distributed according to 4, see [10].

Theorem 1. *Assume H 3-H 5,* $(\beta_1, \beta_2) \in [0,1]$ *and a sequence of decreasing stepsizes* $\{\eta_t\}_{t>0}$, *then the following result holds:*

$$\mathbb{E}\left[\|\nabla f(w_T)\|^2\right] \leq \tilde{C}_1 \sqrt{\frac{d}{T_{\textsf{max}}}} + \tilde{C}_2 \frac{1}{T_{\textsf{max}}} \tag{10}$$

where K is a random termination number distributed according (4) *and the constants are defined as follows:*

$$\begin{aligned} \tilde{C}_1 &= C_1 + \frac{\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\left[\frac{a(1-\beta_1)^2}{1-\beta_2} + 2L\frac{1}{1-\beta_2}\right] \\ C_1 &= \frac{\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\Delta f + \frac{4L\left(\frac{\beta_1}{1-\beta_1}\right)^2 \mathsf{M}}{(1-a\beta_1)+(\beta_1+a)} \frac{(1+\beta_1^2)(1-\beta_1)}{(1-\beta_2)(1-\gamma)} \\ \tilde{C}_2 &= \frac{\mathsf{M}}{(1-\beta_1)\left((1-a\beta_1)+(\beta_1+a)\right)}\tilde{\mathsf{M}}^2 \mathbb{E}\left[\left\|\hat{v}_0^{-1/2}\right\|\right] \end{aligned} \tag{11}$$

See proof in Appendix B.

We remark that the bound for our OPT-AMSGrad method matched the complexity bound of $\mathcal{O}\left(\sqrt{\frac{d}{T_{\textsf{max}}}} + \frac{1}{T_{\textsf{max}}}\right)$ of [10] for SGD and [31] for AMSGrad method.

4.3 Checking H 2 for a Deep Neural Network

We show in this section that the weights satisfy assumption H 2 and stay in a bounded set when the model we are fitting, using our method, is a fully connected feed forward neural network. The activation function for this section will be sigmoid function and we add a ℓ_2 regularization.

For the sake of notation, we assume $\beta_1 = 0$. We consider a fully connected feed forward neural network with L layers modeled by the function $\mathsf{MLN}(w,\xi) : \mathbb{R}^l \to \mathbb{R}$:

$$\mathsf{MLN}(w,\xi) = \sigma\left(w^{(L)}\sigma\left(w^{(L-1)}\ldots\sigma\left(w^{(1)}\xi\right)\right)\right) \tag{12}$$

where $w = [w^{(1)}, w^{(2)}, \cdots, w^{(L)}]$ is the vector of parameters, $\xi \in \mathbb{R}^l$ is the input data and σ is the sigmoid activation function. We assume a l dimension input data and a scalar output for simplicity. The stochastic objective function (3) reads:

$$f(w, \xi) = \mathcal{L}(\mathsf{MLN}(w, \xi), y) + \frac{\lambda}{2} \|w\|^2 \tag{13}$$

where $\mathcal{L}(\cdot, y)$ is the loss function (can be Huber loss or cross entropy), y are the true labels and $\lambda > 0$ is the regularization parameter. For any layer index $\ell \in [1, L]$ we denote the output of layer ℓ by $h^{(\ell)}(w, \xi)$:

$$h^{(\ell)}(w, \xi) = \sigma\left(w^{(\ell)} \sigma\left(w^{(\ell-1)} \dots \sigma\left(w^{(1)} \xi\right)\right)\right)$$

The following Lemma verifies that assumption H 2 is satisfied with a fully connected feed forward neural network (12):

Lemma 4. *Given the multilayer model (12), assume the boundedness of the input data and of the loss function,* i.e., *for any $\xi \in \mathbb{R}^l$ and $y \in \mathbb{R}$ there is a constant $T > 0$ such that $\|\xi\| \le 1$ a.s. and $|\mathcal{L}'(\cdot, y)| \le T$ where $\mathcal{L}'(\cdot, y)$ denotes its derivative* w.r.t. *the parameter. Then for each layer $\ell \in [1, L]$, there exist a constant $A_{(\ell)}$ such that:*

$$\left\|w^{(\ell)}\right\| \le A_{(\ell)}$$

5 Numerical Experiments

5.1 Gradient Estimation

From the analysis in the previous section, we know that whether OPT-AMSGrad converges faster than its counterpart depends on how m_t is chosen. In Optimistic-Online learning, m_t is usually set to $m_t = g_{t-1}$, which means that it uses the previous gradient as a guess of the next one. The choice can accelerate the convergence to equilibrium in some two-player zero-sum games [22, 26, 5], in which each player uses an optimistic online learning algorithm against its opponent.

However, this paper is about solving optimization problems, as in (3), instead of solving zero-sum games. In most classical deep learning tasks, as we will develop in the numerical section, (3) even reads $\min_{w \in \Theta} f(w) = \sum_{i=1}^{n} f(w, \xi_i)$ for a fixed batch of n samples $\{\xi_i\}_{i=1}^{n}$. We propose to use the extrapolation algorithm of [24]. Extrapolation studies estimating the limit of sequence using the last few iterates [2]. Some classical works include Anderson acceleration [28], minimal polynomial extrapolation [3], reduced rank extrapolation [9]. These methods typically assume that the sequence $\{x_t\} \in \mathbb{R}^d$ has a linear relation $x_t = A(x_{t-1} - x^*) + x^*$ and $A \in \mathbb{R}^{d \times d}$ is an unknown, not necessarily symmetric, matrix. The goal is to find the fixed point of x^*. [24] relaxes the assumption to certain degrees. It assumes that the sequence $\{x_t\} \in \mathbb{R}^d$ satisfies

$$x_t - x^* = A(x_{t-1} - x^*) + e_t, \tag{14}$$

where e_t is a second order term satisfying $\|e_t\|_2 = O(\|x_{t-1} - x^*\|_2^2)$ and $A \in \mathbb{R}^{d \times d}$ is an unknown matrix. The extrapolation algorithm we used is shown in Algorithm 3. Some theoretical guarantees regarding the distance between the output and x^* are provided in [24]. For our numerical experiments in the next section, we run OPT-AMSGrad using Algorithm 3 to get m_t. Specifically, m_t is obtained by (a) calling Algorithm 3 with input being a sequence of some past $r+1$ gradients, $\{g_t, g_{t-1}, g_{t-2}, \dots, g_{t-r}\}$, where r is a parameter and (b) setting $m_t := \Sigma_{i=0}^{r-1} c_i g_{t-r+i}$ from the output of Algorithm 3. To see why the output from the extrapolation method may be a reasonable

Algorithm 3 Regularized Approximate Minimal Polynomial Extrapolation (RMPE) [24]

```
1: Input: sequence $\{x_s \in \mathbb{R}^d\}_{s=0}^{s=r}$, parameter $\lambda > 0$.
2: Compute matrix $U = [x_1 - x_0, \dots, x_r - x_{r-1}] \in \mathbb{R}^{d \times r}$.
3: Obtain $z$ by solving $(U^\top U + \lambda I)z = \mathbf{1}$.
4: Get $c = z/(z^\top \mathbf{1})$.
5: Output: $\Sigma_{i=0}^{r-1} c_i x_i$, the approximation of the fixed point $x^*$.
```

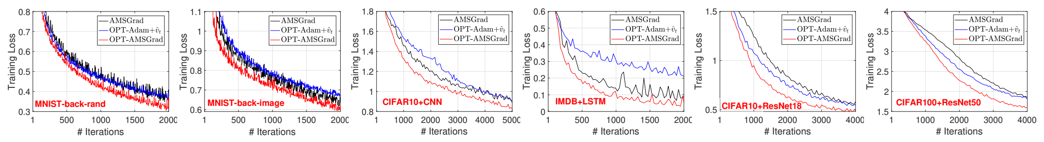


Figure 2: Training loss vs. Number of iterations. The first row are results with fully-connected NN.

estimation, assume that the update converges to a stationary point (i.e. $g^* := \nabla f(w^*) = 0$ for the underlying function f). Then, we might rewrite (14) as

$$g_t = A g_{t-1} + O(\|g_{t-1}\|_2^2) u_{t-1}, \tag{15}$$

for some vector u_{t-1} with a unit norm. The equation suggests that the next gradient vector g_t is a linear transform of g_{t-1} plus an error vector that may not be in the span of A whose length is $O(\|g_{t-1}\|_2^2)$. If the algorithm is guaranteed to converge to a stationary point, the magnitude of the error component will eventually go to zero. We remark that the choice of algorithm for gradient prediction is surely not unique. We propose to use the recent result among various related works. Indeed, one can use any method that can provide reasonable guess of gradient in next iteration.

5.2 Classification Experiments

In this section, we provide experiments on classification tasks with various neural network architectures and datasets to demonstrate the effectiveness of OPTIMISTIC-AMSGRAD in practical applications.

Methods. We consider two baselines. The first one is the original AMSGRAD. The hyper-parameters are set to be β_1 and β_2 to be 0.9 and 0.999 respectively, as recommended by [23]. We tune the learning rate η over a fine grid and report the best result. The other competing method is the aforementioned OPTIMISTIC-ADAM+$\hat{v}_t$ method (Algorithm **??**) as in Section 3. The key difference is that it uses previous gradient as the gradient prediction of the next iteration. We also report the best result achieved by tuning the step size η for OPTIMISTIC-ADAM+$\hat{v}_t$. For OPTIMISTIC-AMSGRAD, we use the same β_1, β_2 and the best step size η of AMSGRAD for a fair evaluation of the improvement brought by the extra optimistic step. Yet, OPTIMISTIC-AMSGRAD has an additional parameter r that controls the number of previous gradients used for gradient prediction. Fortunately, we observe similar performance of OPTIMISTIC-AMSGRAD with different values of r. Hence, we report $r = 5$ for now when comparing with other baselines. We will address on the choice of r at the end of this section. In all experiments, all the optimization algorithms are initialized at the same point. We report the results averaged over 5 repetitions.

Datasets. Following [23] and [15], we compare different algorithms on *MNIST*, *CIFAR10*, *CIFAR100*, and *IMDB* datasets. For *MNIST*, we use two noisy variants named as 1.65*MNIST-back-rand* and 1.65*MNIST-back-image* from [16]. They both have 12000 training samples and 50000 test samples, where random background is inserted to the original *MNIST* hand written digit images. For *MNIST-back-rand*, each image is inserted with a random background, whose pixel values generated uniformly from 0 to 255, while *MNIST-back-image* takes random patches from a black and white as noisy background. The input dimension is 784 (28×28) and the number of classes is 10. *CIFAR10* and *CIFAR100* are popular computer-vision datasets consisting of 50000 training images and 10000 test images, of size 32×32. The number of classes are 10 and 100, respectively. The *IMDB* movie review dataset is a binary classification dataset with 25000 training and testing samples respectively. It is a popular datasets for text classification.

Network architecture. We adopt a multi-layer fully-connected neural network with input layer followed by a hidden layer with 200 nodes, which is connected to another layer with 100 nodes before the output layer. The activation function is ReLU for hidden layers, and softmax for the output layer. This network is tested on *MNIST* variants. Since convolutional networks are popular for image classification tasks, we consider an ALL-CNN architecture proposed by [25], which is constructed with several convolutional blocks and dropout layers. In addition, we also apply residual networks, Resnet-18 and Resnet-50 [14], which have achieved many state-of-the-art results. For the texture *IMDB* dataset, we consider training a Long-Short Term Memory (LSTM) network. The network includes a word embedding layer with 5000 input entries representing most frequent words

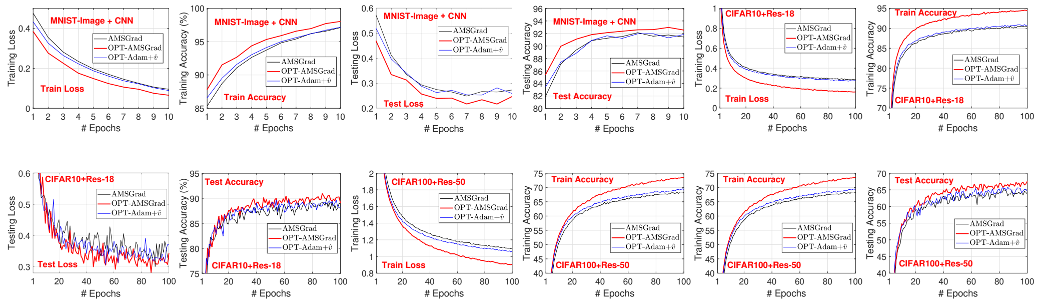
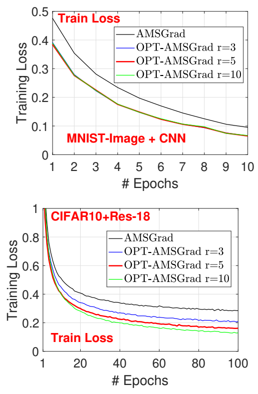


Figure 3: *MNIST-back-image* + CNN, *CIFAR10* + Res-18 and *CIFAR100* + Res-50 . We compare three methods in terms of training (cross-entropy) loss and accuracy, testing loss and accuracy.

in the dataset, and each word is embedded into a 32 dimensional space. The output of the embedding layer is passed to 100 LSTM units, which is then connected to 100 fully connected ReLu's before the output layer. For all the models, we use cross-entropy loss. A mini-batch size of 128 is used to compute the stochastic gradients.

Results. Firstly, to illustrate the acceleration effect of OPTIMISTIC-AMSGRAD at early stage, we provide the training loss against number of iterations in Figure 2. We clearly observe that on all datasets, the proposed OPTIMISTIC-AMSGRAD converges faster than the other competing methods, right after the training begins. In other words, we need fewer iterations (samples) to achieve the same training loss. This validates one of the main advantages of OPTIMISTIC-AMSGRAD, which is a higher sample efficiency. We are also curious about the long-term performance and generalization of the proposed method in test phase. In Figure 3, we plot the corresponding results when the model is trained to the state with stable test accuracy. We observe: 1) In the long term, OPTIMISTIC-AMSGRAD algorithm may converge to a better point with smaller objective function value, and 2) In this three applications, the proposed OPTIMISTIC-AMSGRAD also outperforms the competing methods in terms of test accuracy. These are also important benefits of OPTIMISTIC-AMSGRAD.

5.3 Choice of parameter r



Recall that our proposed algorithm has the parameter r that governs the use of past information. Figure 4 compares the performance under different values of $r = 3, 5, 10$ on two datasets. From the result we see that the choice of r does not have significant impact on learning performance. Taking into consideration both quality of gradient prediction and computational cost, it appears that $r = 5$ is a good choice. We remark that empirically, the performance comparison among $r = 3, 5, 10$ is not absolutely consistent (i.e. more means better) in all cases. One possible reason is that for deep neural nets which have very complicated and highly non-convex landscape, using gradient information from too long ago may not be helpful in accurate gradient prediction. Nevertheless, $r = 5$ seems to be good for most applications.

6 Conclusion

In this paper, we propose OPTIMISTIC-AMSGRAD, which combines optimistic learning and AMSGRAD to improve sampling efficiency and accelerate the process of training, in particular for deep neural networks. With a good gradient prediction, the regret can be smaller than that of standard AMSGRAD. Experiments on various deep learning problems demonstrate the effectiveness of the proposed method in improving the training efficiency.

7 To ADD?

7.1 Choice of parameter r

Recall that our proposed algorithm has the parameter r that governs the use of past information. Figure 4 compares the performance under different values of $r = 3, 5, 10$ on two datasets. From the result we see that the choice of r does not have significant impact on learning performance. Taking into consideration both quality of gradient prediction and computational cost, it appears that $r = 5$ is a good choice. We remark that empirically, the performance comparison among $r = 3, 5, 10$ is not absolutely consistent (i.e. more means better) in all cases. One possible reason is that for deep neural nets which have very complicated and highly non-convex landscape, using gradient information from too long ago may not be helpful in accurate gradient prediction. Nevertheless, $r = 5$ seems to be good for most applications.

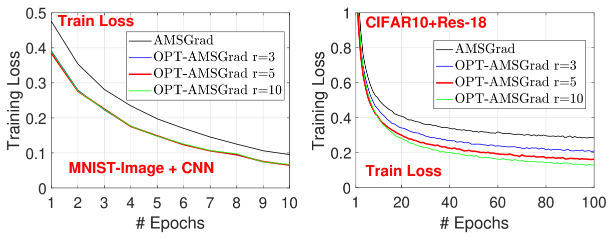


Figure 4: The training loss of OPTIMISTIC-AMSGRAD with different r.

7.2 Concluding Remarks

Discussion on the iteration cost

We observe that the iteration cost (i.e., actual running time per iteration) of our implementation of OPTIMISTIC-AMSGRAD with $r = 5$ is roughly two times larger than the standard AMSGRAD. When $r = 3$, the cost is roughly 0.7 times longer. Nevertheless, OPTIMISTIC-AMSGRAD may still be beneficial in terms of training efficiency, since fewer iterations are typically needed. For example, in Figure **??** and **??**, to reach the training loss of AMSGRAD at 100 epochs, the proposed method only needs roughly 20 and 40 epochs, respectively. That said, OPTIMISTIC-AMSGRAD needs 40% and 80% time to achieve same training loss as AMSGrad, in this two problems.

The computational overhead mostly comes from the gradient extrapolation step. More specifically, recall that the extrapolation step consists of: (a) The step of constructing the linear system $(U^\top U)$. The cost of this step can be optimized and reduced to $\mathcal{O}(d)$, since the matrix U only changes one column at a time. (b) The step of solving the linear system. The cost of this step is $O(r^3)$, which is negligible as the linear system is very small (5-by-5 if $r = 5$). (c) The step that outputs an estimated gradient as a weighted average of previous gradients. The cost of this step is $\mathcal{O}(r \times d)$. Thus, the computational overhead is $\mathcal{O}\left((r + 1)d + r^3\right)$. Yet, we notice that step (a) and (c) is parallelizable, so they can be accelerated in practice.

Memory usage: Our algorithm needs a storage of past r gradients for each coordinate, in addition to the estimated second moments and the moving average. Though it seems demanding compared to the standard AMSGrad, it is relatively cheap compared to Natural gradient method (e.g., [**?**]), as Natural gradient method needs to store some matrix inverse.

References

[1] J. Abernethy, K. A. Lai, K. Y. Levy, and J.-K. Wang. Faster rates for convex-concave games. *COLT*, 2018.

[2] C. Brezinski and M. R. Zaglia. Extrapolation methods: theory and practice. *Elsevier*, 2013.

[3] S. Cabay and L. Jackson. A polynomial extrapolation method for finding limits and antilimits of vector sequences. *SIAM Journal on Numerical Analysis*, 1976.

[4] C.-K. Chiang, T. Yang, C.-J. Lee, M. Mahdavi, C.-J. Lu, R. Jin, and S. Zhu. Online optimization with gradual variations. *COLT*, 2012.

[5] C. Daskalakis, A. Ilyas, V. Syrgkanis, and H. Zeng. Training gans with optimism. *ICLR*, 2018.

[6] A. Défossez, L. Bottou, F. Bach, and N. Usunier. On the convergence of adam and adagrad. *arXiv preprint arXiv:2003.02395*, 2020.

[7] T. Dozat. Incorporating nesterov momentum into adam. *ICLR (Workshop Track)*, 2016.

[8] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)*, 2011.

[9] R. Eddy. Extrapolating to the limit of a vector sequence. *Information linkage between applied mathematics and industry, Elsevier*, 1979.

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

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *NIPS*, 2014.

[12] A. Graves, A. rahman Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. *ICASSP*, 2013.

[13] E. Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2016.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, 2016.

[15] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[16] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. *ICML*, 2007.

[17] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *NIPS*, 2017.

[18] H. B. McMahan and M. J. Streeter. Adaptive bound optimization for online convex optimization. *COLT*, 2010.

[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *NIPS (Deep Learning Workshop)*, 2013.

[20] Y. Nesterov. Introductory lectures on convex optimization: A basic course. *Springer*, 2004.

[21] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *Mathematics and Mathematical Physics*, 1964.

[22] A. Rakhlin and K. Sridharan. Optimization, learning, and games with predictable sequences. *NIPS*, 2013.

[23] S. J. Reddi, S. Kale, and S. Kumar. On the convergence of adam and beyond. *ICLR*, 2018.

[24] D. Scieur, A. d'Aspremont, and F. Bach. Regularized nonlinear acceleration. *NIPS*, 2016.

[25] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *ICLR*, 2015.

[26] V. Syrgkanis, A. Agarwal, H. Luo, and R. E. Schapire. Fast convergence of regularized learning in games. *NIPS*, 2015.

[27] T. Tieleman and G. Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 2012.

[28] H. F. Walker and P. Ni. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis*, 2011.

[29] Y. Yan, T. Yang, Z. Li, Q. Lin, and Y. Yang. A unified analysis of stochastic momentum methods for deep learning. *arXiv preprint arXiv:1808.10396*, 2018.

[30] M. D. Zeiler. Adadelta: An adaptive learning rate method. *arXiv:1212.5701*, 2012.

[31] D. Zhou, Y. Tang, Z. Yang, Y. Cao, and Q. Gu. On the convergence of adaptive gradient methods for nonconvex optimization. *arXiv preprint arXiv:1808.05671*, 2018.

A Proofs of Auxiliary Lemmas

A.1 Proof of Lemma 1

Lemma. *Assume assumption H 5, then the quantities defined in Algorithm 2 satisfy for any $w \in \Theta$ and $t > 0$:*

$$\|\nabla f(w_t)\| < \mathsf{M}, \quad \|\theta_t\| < \mathsf{M}, \quad \|\hat{v}_t\| < \mathsf{M}^2 \ .$$

Proof Assume assumption H 5 we have:

$$\|\nabla f(w)\| = \|\mathbb{E}[\nabla f(w, \xi)]\| \le \mathbb{E}[\|\nabla f(w, \xi)\|] \le \mathsf{M}$$

By induction reasoning, since $\|\theta_0\| = 0 \le \mathsf{M}$ and suppose that for $\|\theta_t\| \le \mathsf{M}$ then we have

$$\|\theta_{t+1}\| = \|\beta_1 \theta_t + (1-\beta_1)\, g_{t+1}\| \le \beta_1 \|\theta_t\| + (1-\beta_1)\, \|g_{t+1}\| \le \mathsf{M} \tag{16}$$

Using the same induction reasoning we prove that

$$\|\hat{v}_{t+1}\| = \left\|\beta_2 \hat{v}_t + (1-\beta_2)\, g_{t+1}^2\right\| \le \beta_2 \|\hat{v}_t\| + (1-\beta_1)\, \left\|g_{t+1}^2\right\| \le \mathsf{M}^2 \tag{17}$$

□

A.2 Proof of Lemma 2

Lemma. *Assume a strictly positive and non increasing sequence of stepsizes $\{\eta_t\}_{t>0}$, $\beta_\in[0, 1]$, then the following holds:*

$$\overline{w}_{t+1} - \overline{w}_t \le \frac{\beta_1}{1-\beta_1} \tilde{\theta}_{t-1} \left[\eta_{t-1} \hat{v}_{t-1}^{-1/2} - \eta_t \hat{v}_t^{-1/2}\right] - \eta_t \hat{v}_t^{-1/2} \tilde{g}_t \ , \tag{18}$$

where $\tilde{\theta}_t = \theta_t + \beta_1 \theta_{t-1}$ and $\tilde{g}_t = g_t - \beta_1 m_t + \beta_1 g_{t-1} + m_{t+1}$.

Proof By definition (7) and using the Algorithm updates, we have:

$$\begin{aligned}
\overline{w}_{t+1} - \overline{w}_t &= \frac{1}{1-\beta_1}(w_{t+1} - w_t) - \frac{\beta_1}{1-\beta_1}(w_t - w_{t-1}) \\
&= -\frac{1}{1-\beta_1}\eta_t \hat{v}_t^{-1/2}(\theta_t + h_{t+1}) + \frac{\beta_1}{1-\beta_1}\eta_{t-1}\hat{v}_{t-1}^{-1/2}(\theta_{t-1} + h_t) \\
&= -\frac{1}{1-\beta_1}\eta_t \hat{v}_t^{-1/2}(\theta_t + \beta_1\theta_{t-1}) - \frac{1}{1-\beta_1}\eta_t \hat{v}_t^{-1/2}(1-\beta_1) m_{t+1} \\
&\quad + \frac{\beta_1}{1-\beta_1}\eta_{t-1}\hat{v}_{t-1}^{-1/2}(\theta_{t-1} + \beta_1\theta_{t-2}) + \frac{\beta_1}{1-\beta_1}\eta_{t-1}\hat{v}_{t-1}^{-1/2}(1-\beta_1) m_t
\end{aligned} \tag{19}$$

Denote $\tilde{\theta}_t = \theta_t + \beta_1 \theta_{t-1}$ and $\tilde{g}_t = g_t - \beta_1 m_t + \beta_1 g_{t-1} + m_{t+1}$. Notice that $\tilde{\theta}_t = \beta_1 \tilde{\theta}_{t-1} + (1-\beta_1)(g_t + \beta_1 g_{t-1})$.

$$\overline{w}_{t+1} - \overline{w}_t \le \frac{\beta_1}{1-\beta_1} \tilde{\theta}_{t-1} \left[\eta_{t-1} \hat{v}_{t-1}^{-1/2} - \eta_t \hat{v}_t^{-1/2}\right] - \eta_t \hat{v}_t^{-1/2} \tilde{g}_t \tag{20}$$

□

A.3 Proof of Lemma 3

Lemma. *Assume H 5, a strictly positive and a sequence of constant stepsizes $\{\eta_t\}_{t>0}$, $\beta_\in[0, 1]$, then the following holds:*

$$\sum_{t=1}^{T_{\mathsf{max}}} \eta_t^2 \mathbb{E}\left[\left\|\hat{v}_t^{-1/2}\theta_t\right\|_2^2\right] \le \frac{\eta^2 d T_{\mathsf{max}}(1-\beta_1)}{(1-\beta_2)(1-\gamma)} \tag{21}$$

Proof We denote by index $p \in [1, d]$ the dimension of each component of vectors of interest. Noting that for any $t > 0$ and dimension p we have $\hat{v}_{t,p} \geq v_{t,p}$, then:

$$\begin{aligned}
\eta_t^2 \mathbb{E}\left[\left\|\hat{v}_t^{-1/2}\theta_t\right\|_2^2\right] &= \eta_t^2 \mathbb{E}\left[\sum_{p=1}^{d} \frac{\theta_{t,p}^2}{\hat{v}_{t,p}}\right] \\
&\leq \eta_t^2 \mathbb{E}\left[\sum_{i=1}^{d} \frac{\theta_{t,p}^2}{v_{t,p}}\right] \\
&\leq \eta_t^2 \mathbb{E}\left[\sum_{i=1}^{d} \frac{(\sum_{r=1}^{t}(1-\beta_1)\beta_1^{t-r} g_{r,p})^2}{\sum_{r=1}^{t}(1-\beta_2)\beta_2^{t-r} g_{r,p}^2}\right]
\end{aligned} \tag{22}$$

where the last inequality is due to initializations. Denote $\gamma = \frac{\beta_1}{\beta_2}$. Then,

$$\begin{aligned}
\eta_t^2 \mathbb{E}\left[\left\|\hat{v}_t^{-1/2}\theta_t\right\|_2^2\right] &\leq \frac{\eta_t^2(1-\beta_1)^2}{1-\beta_2} \mathbb{E}\left[\sum_{i=1}^{d} \frac{(\sum_{r=1}^{t}\beta_1^{t-r} g_{r,p})^2}{\sum_{r=1}^{t}\beta_2^{t-r} g_{r,p}^2}\right] \\
&\overset{(a)}{\leq} \frac{\eta_t^2(1-\beta_1)}{1-\beta_2} \mathbb{E}\left[\sum_{i=1}^{d} \frac{\sum_{r=1}^{t}\beta_1^{t-r} g_{r,p}^2}{\sum_{r=1}^{t}\beta_2^{t-r} g_{r,p}^2}\right] \\
&\leq \frac{\eta_t^2(1-\beta_1)}{1-\beta_2} \mathbb{E}\left[\sum_{i=1}^{d}\sum_{r=1}^{t} \gamma^{t-r}\right] = \frac{\eta_t^2 d(1-\beta_1)}{1-\beta_2} \mathbb{E}\left[\sum_{r=1}^{t} \gamma^{t-r}\right]
\end{aligned} \tag{23}$$

where (a) is due to $\sum_{r=1}^{t} \beta_1^{t-r} \leq \frac{1}{1-\beta_1}$. Summing from $t = 1$ to $t = T_{\mathsf{max}}$ on both sides yields:

$$\begin{aligned}
\sum_{t=1}^{T_{\mathsf{max}}} \eta_t^2 \mathbb{E}\left[\left\|\hat{v}_t^{-1/2}\theta_t\right\|_2^2\right] &\leq \frac{\eta_t^2 d(1-\beta_1)}{1-\beta_2} \mathbb{E}\left[\sum_{t=1}^{T_{\mathsf{max}}}\sum_{r=1}^{t} \gamma^{t-r}\right] \\
&\leq \frac{\eta^2 dT(1-\beta_1)}{1-\beta_2} \mathbb{E}\left[\sum_{t=t}^{t} \gamma^{t-r}\right] \\
&\leq \frac{\eta^2 dT(1-\beta_1)}{(1-\beta_2)(1-\gamma)}
\end{aligned} \tag{24}$$

where the last inequality is due to $\sum_{r=1}^{t} \gamma^{t-r} \leq \frac{1}{1-\gamma}$ by definition of γ. □

B Proofs of Theorem 1

Theorem. *Assume H 3-H 5, $(\beta_1, \beta_2) \in [0, 1]$ and a sequence of decreasing stepsizes $\{\eta_t\}_{t>0}$, then the following result holds:*

$$\mathbb{E}\left[\|\nabla f(w_T)\|^2\right] \leq \tilde{C}_1 \sqrt{\frac{d}{T_{\mathsf{max}}}} + \tilde{C}_2 \frac{1}{T_{\mathsf{max}}} \tag{25}$$

where T is a random termination number distributed according (4) and the constants are defined as follows:

$$\begin{aligned}
\tilde{C}_1 &= C_1 + \frac{\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\left[\frac{a(1-\beta_1)^2}{1-\beta_2} + 2L\frac{1}{1-\beta_2}\right] \\
C_1 &= \frac{\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\Delta f + \frac{4L\left(\frac{\beta_1}{1-\beta_1}\right)^2 \mathsf{M}}{(1-a\beta_1)+(\beta_1+a)} \frac{(1+\beta_1^2)(1-\beta_1)}{(1-\beta_2)(1-\gamma)} \\
\tilde{C}_2 &= \frac{\mathsf{M}}{(1-\beta_1)((1-a\beta_1)+(\beta_1+a))} \tilde{\mathsf{M}}^2 \mathbb{E}\left[\left\|\hat{v}_0^{-1/2}\right\|\right]
\end{aligned} \tag{26}$$

Proof Using H 3 and the iterate $\overline{w}_t$ we have:

$$\begin{aligned} f(\overline{w}_{t+1}) &\leq f(\overline{w}_t) + \nabla f(\overline{w}_t)^\top (\overline{w}_{t+1} - \overline{w}_t) + \frac{L}{2}\left\|\overline{w}_{t+1} - \overline{w}_t\right\|^2 \\ &\leq f(\overline{w}_t) + \underbrace{\nabla f(w_t)^\top (\overline{w}_{t+1} - \overline{w}_t)}_{A} + \underbrace{\left(\nabla f(\overline{w}_t) - \nabla f(w_t)\right)^\top (\overline{w}_{t+1} - \overline{w}_t)}_{B} + \frac{L}{2}\left\|\overline{w}_{t+1} - \overline{w}_t\right\| \end{aligned} \tag{27}$$

Term A. Using Lemma 2, we have that:

$$\begin{aligned} \nabla f(w_t)^\top (\overline{w}_{t+1} - \overline{w}_t) &\leq \nabla f(w_t)^\top \left[\frac{\beta_1}{1-\beta_1}\tilde{\theta}_{t-1}\left[\eta_{t-1}\hat{v}_{t-1}^{-1/2} - \eta_t\hat{v}_t^{-1/2}\right] - \eta_t\hat{v}_t^{-1/2}\tilde{g}_t\right] \\ &\leq \frac{\beta_1}{1-\beta_1}\left\|\nabla f(w_t)\right\|\left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2} - \eta_t\hat{v}_t^{-1/2}\right\|\left\|\tilde{\theta}_{t-1}\right\| - \nabla f(w_t)^\top \eta_t\hat{v}_t^{-1/2}\tilde{g}_t \end{aligned} \tag{28}$$

where the inequality is due to trivial inequality for positive diagonal matrix. Using Lemma 1 and assumption H4 we obtain:

$$\nabla f(w_t)^\top (\overline{w}_{t+1} - \overline{w}_t) \leq \frac{\beta_1(1+\beta_1)}{1-\beta_1}\mathsf{M}^2\left[\left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}\right\| - \left\|\eta_t\hat{v}_t^{-1/2}\right\|\right] - \nabla f(w_t)^\top \eta_t\hat{v}_t^{-1/2}\tilde{g}_t \tag{29}$$

where we have used the fact that $\eta_t\hat{v}_t^{-1/2}$ is a diagonal matrix such that $\eta_{t-1}\hat{v}_{t-1}^{-1/2} \succcurlyeq \eta_t\hat{v}_t^{-1/2} \succcurlyeq 0$ (decreasing stepsize and max operator). Also note that:

$$\begin{aligned} -\nabla f(w_t)^\top \eta_t\hat{v}_t^{-1/2}\tilde{g}_t &= -\nabla f(w_t)^\top \eta_{t-1}\hat{v}_{t-1}^{-1/2}\bar{g}_t - \nabla f(w_t)^\top\left[\eta_t\hat{v}_t^{-1/2} - \eta_t\hat{v}_t^{-1/2}\right]\bar{g}_t \\ &\quad - \nabla f(w_t)^\top \eta_{t-1}\hat{v}_{t-1}^{-1/2}(\beta_1 g_{t-1} + m_{t+1}) \\ &\leq -\nabla f(w_t)^\top \eta_{t-1}\hat{v}_{t-1}^{-1/2}\bar{g}_t + (1 - a\beta_1)\mathsf{M}^2\left[\left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}\right\| - \left\|\eta_t\hat{v}_t^{-1/2}\right\|\right] \\ &\quad - \nabla f(w_t)^\top \eta_t\hat{v}_t^{-1/2}(\beta_1 g_{t-1} + m_{t+1}) \end{aligned} \tag{30}$$

using Lemma 1 on $\|g_t\|$ and where that $\tilde{g}_t = \bar{g}_t + \beta_1 g_{t-1} + m_{t+1} = g_t - \beta_1 m_t + \beta_1 g_{t-1} + m_{t+1}$. Plugging (30) into (29) yields:

$$\begin{aligned} &\nabla f(w_t)^\top (\overline{w}_{t+1} - \overline{w}_t) \\ &\leq -\nabla f(w_t)^\top \eta_{t-1}\hat{v}_{t-1}^{-1/2}\bar{g}_t + \frac{1}{1-\beta_1}(a\beta_1^2 - 2a\beta_1 + \beta 1)\mathsf{M}^2\left[\left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}\right\| - \left\|\eta_t\hat{v}_t^{-1/2}\right\|\right] \\ &- \nabla f(w_t)^\top \eta_t\hat{v}_t^{-1/2}(\beta_1 g_{t-1} + m_{t+1}) \end{aligned} \tag{31}$$

Term B. By Cauchy-Schwarz (CS) inequality we have:

$$\left(\nabla f(\overline{w}_t) - \nabla f(w_t)\right)^\top (\overline{w}_{t+1} - \overline{w}_t) \leq \left\|\nabla f(\overline{w}_t) - \nabla f(w_t)\right\|\left\|\overline{w}_{t+1} - \overline{w}_t\right\| \tag{32}$$

Using smoothness assumption H 3:

$$\begin{aligned} \left\|\nabla f(\overline{w}_t) - \nabla f(w_t)\right\| &\leq L\left\|\overline{w}_t - w_t\right\| \\ &\leq L\frac{\beta_1}{1-\beta_1}\left\|w_t - w_{t-1}\right\| \end{aligned} \tag{33}$$

By Lemma 2 we also have:

$$\begin{aligned} \overline{w}_{t+1} - \overline{w}_t &= \frac{\beta_1}{1-\beta_1}\tilde{\theta}_{t-1}\left[\eta_{t-1}\hat{v}_{t-1}^{-1/2} - \eta_t\hat{v}_t^{-1/2}\right] - \eta_t\hat{v}_t^{-1/2}\tilde{g}_t \\ &= \frac{\beta_1}{1-\beta_1}\tilde{\theta}_{t-1}\eta_{t-1}\hat{v}_{t-1}^{-1/2}\left[I - (\eta_t\hat{v}_t^{-1/2})(\eta_{t-1}\hat{v}_{t-1}^{-1/2})^{-1}\right] - \eta_t\hat{v}_t^{-1/2}\tilde{g}_t \\ &= \frac{\beta_1}{1-\beta_1}\left[I - (\eta_t\hat{v}_t^{-1/2})(\eta_{t-1}\hat{v}_{t-1}^{-1/2})^{-1}\right](w_{t-1} - w_t) - \eta_t\hat{v}_t^{-1/2}\tilde{g}_t \end{aligned} \tag{34}$$

where the last equality is due to $\tilde{\theta}_{t-1}\eta_{t-1}\hat{v}_{t-1}^{-1/2} = w_{t-1} - w_t$ by construction of $\tilde{\theta}_t$. Taking the norms on both sides, observing $\left\| I - (\eta_t \hat{v}_t^{-1/2})(\eta_{t-1}\hat{v}_{t-1}^{-1/2})^{-1} \right\| \leq 1$ due to the decreasing stepsize and the construction of $\hat{v}_t$ and using CS inequality yield:

$$\|\overline{w}_{t+1} - \overline{w}_t\| \leq \frac{\beta_1}{1-\beta_1}\|w_{t-1} - w_t\| + \left\|\eta_t \hat{v}_t^{-1/2}\tilde{g}_t\right\| \tag{35}$$

We recall Young's inequality with a constant $\delta \in (0,1)$ as follows:

$$\langle X \,|\, Y\rangle \leq \frac{1}{\delta}\|X\|^2 + \delta\|Y\|^2$$

Plugging (33) and (35) into (32) returns:

$$\begin{aligned}(\nabla f(\overline{w}_t) - \nabla f(w_t))^\top (\overline{w}_{t+1} - \overline{w}_t) \leq & L\frac{\beta_1}{1-\beta_1}\left\|\eta_t \hat{v}_t^{-1/2}\tilde{g}_t\right\| \|w_t - w_{t-1}\| \\ & + L\left(\frac{\beta_1}{1-\beta_1}\right)^2 \|w_{t-1} - w_t\|^2\end{aligned} \tag{36}$$

Applying Young's inequality with $\delta \to \frac{\beta_1}{1-\beta_1}$ on the product $\left\|\eta_t \hat{v}_t^{-1/2}\tilde{g}_t\right\| \|w_t - w_{t-1}\|$ yields:

$$(\nabla f(\overline{w}_t) - \nabla f(w_t))^\top (\overline{w}_{t+1} - \overline{w}_t) \leq L\left\|\eta_t \hat{v}_t^{-1/2}\tilde{g}_t\right\|^2 + 2L\left(\frac{\beta_1}{1-\beta_1}\right)^2 \|w_{t-1} - w_t\|^2 \tag{37}$$

The last term $\frac{L}{2}\|\overline{w}_{t+1} - \overline{w}_t\|$ can be upper bounded using (35):

$$\begin{aligned}\frac{L}{2}\|\overline{w}_{t+1} - \overline{w}_t\|^2 &\leq \frac{L}{2}\left[\frac{\beta_1}{1-\beta_1}\|w_{t-1} - w_t\| + \left\|\eta_t \hat{v}_t^{-1/2}\tilde{g}_t\right\|\right] \\ &\leq L\left\|\eta_t \hat{v}_t^{-1/2}\tilde{g}_t\right\|^2 + 2L\left(\frac{\beta_1}{1-\beta_1}\right)^2 \|w_{t-1} - w_t\|^2\end{aligned} \tag{38}$$

Plugging (31), (37) and (38) into (27) and taking the expectations on both sides give:

$$\begin{aligned}&\mathbb{E}\left[f(\overline{w}_{t+1}) + \frac{1}{1-\beta_1}\tilde{\mathsf{M}}^2\left\|\eta_t \hat{v}_t^{-1/2}\right\| - \left(f(\overline{w}_t) + \frac{1}{1-\beta_1}\tilde{\mathsf{M}}^2\left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}\right\|\right)\right] \\ &\leq \mathbb{E}\left[-\nabla f(w_t)^\top \eta_{t-1}\hat{v}_{t-1}^{-1/2}\bar{g}_t - \nabla f(w_t)^\top \eta_t \hat{v}_t^{-1/2}(\beta_1 g_{t-1} + m_{t+1})\right] \\ &+ \mathbb{E}\left[2L\left\|\eta_t \hat{v}_t^{-1/2}\tilde{g}_t\right\|^2 + 4L\left(\frac{\beta_1}{1-\beta_1}\right)^2 \|w_{t-1} - w_t\|^2\right]\end{aligned} \tag{39}$$

where $\tilde{\mathsf{M}}^2 = (a\beta_1^2 - 2a\beta_1 + \beta 1)\mathsf{M}^2$. Note that the expectation of $\tilde{g}_t$ conditioned on the filtration $\mathcal{F}_t$ reads as follows

$$\begin{aligned}\mathbb{E}\left[\nabla f(w_t)^\top \bar{g}_t\right] &= \mathbb{E}\left[\nabla f(w_t)^\top (g_t - \beta_1 m_t)\right] \\ &= (1 - a\beta_1)\|\nabla f(w_t)\|^2\end{aligned} \tag{40}$$

Summing from $t = 1$ to $t = T$ leads to

$$\begin{aligned}&\frac{1}{\mathsf{M}}\sum_{t=1}^{T_{\max}}((1-a\beta_1)\eta_{t-1} + (\beta_1 + a)\eta_t)\|\nabla f(w_t)\|^2 \leq \\ &\mathbb{E}\left[f(\overline{w}_1) + \frac{1}{1-\beta_1}\tilde{\mathsf{M}}^2\left\|\eta_0\hat{v}_0^{-1/2}\right\| - \left(f(\overline{w}_{T_{\max}+1}) + \frac{1}{1-\beta_1}\tilde{\mathsf{M}}^2\left\|\eta_{T_{\max}}\hat{v}_{T_{\max}}^{-1/2}\right\|\right)\right] \\ &+ 2L\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\left\|\eta_t \hat{v}_t^{-1/2}\tilde{g}_t\right\|^2\right] + 4L\left(\frac{\beta_1}{1-\beta_1}\right)^2\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\|w_{t-1} - w_t\|^2\right] \\ &\leq \mathbb{E}\left[\Delta f + \frac{1}{1-\beta_1}\tilde{\mathsf{M}}^2\left\|\eta_0\hat{v}_0^{-1/2}\right\|\right] + 2L\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\left\|\eta_t \hat{v}_t^{-1/2}\tilde{g}_t\right\|^2\right] + 4L\left(\frac{\beta_1}{1-\beta_1}\right)^2\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\|w_{t-1} - w_t\|^2\right]\end{aligned} \tag{41}$$

where $\Delta f = f(\overline{w}_1) - f(\overline{w}_{T_{\max}+1})$. We note that by definition of $\hat{v}_t$, and a constant learning rate η_t, we have

$$\begin{aligned}\|w_{t-1} - w_t\|^2 &= \left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}(\theta_{t-1} + h_t)\right\|^2 \\ &= \left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}(\theta_{t-1} + \beta_1\theta_{t-2} + (1-\beta_1)m_t)\right\|^2 \\ &\leq \left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}\theta_{t-1}\right\|^2 + \left\|\eta_{t-2}\hat{v}_{t-2}^{-1/2}\beta_1\theta_{t-2}\right\|^2 + (1-\beta_1)^2\left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}m_t\right\|^2\end{aligned} \tag{42}$$

Using Lemma 3 we have

$$\begin{aligned}&\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\|w_{t-1} - w_t\|^2\right] \\ &\leq (1+\beta_1^2)\frac{\eta^2 d T_{\max}(1-\beta_1)}{(1-\beta_2)(1-\gamma)} + (1-\beta_1)^2\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}m_t\right\|^2\right]\end{aligned} \tag{43}$$

And thus, setting the learning rate to a constant value η and injecting in (41) yields:

$$\begin{aligned}&\mathbb{E}\left[\|\nabla f(w_T)\|^2\right] = \frac{1}{\sum_{j=1}^{T_{\max}}\eta_j}\sum_{t=1}^{T_{\max}}\eta_t\|\nabla f(w_t)\|^2 \\ &\leq \frac{\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\frac{1}{\sum_{j=1}^{T_{\max}}\eta_j}\mathbb{E}\left[\Delta f + \frac{1}{1-\beta_1}\tilde{\mathsf{M}}^2\left\|\eta_0\hat{v}_0^{-1/2}\right\|\right] \\ &+ \frac{4L\left(\frac{\beta_1}{1-\beta_1}\right)^2\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\frac{1}{\sum_{j=1}^{T_{\max}}\eta_j}(1+\beta_1^2)\frac{\eta^2 d T_{\max}(1-\beta_1)}{(1-\beta_2)(1-\gamma)} \\ &+ \frac{\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\frac{1}{\sum_{j=1}^{T_{\max}}\eta_j}(1-\beta_1)^2\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\left\|\eta_{t-1}\hat{v}_{t-1}^{-1/2}m_t\right\|^2\right] \\ &+ \frac{2L\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\frac{1}{\sum_{j=1}^{T_{\max}}\eta_j}\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\left\|\eta_t\hat{v}_t^{-1/2}\tilde{g}_t\right\|^2\right]\end{aligned} \tag{44}$$

where T is a random termination number distributed according (4). Setting the stepsize to $\eta = \frac{1}{\sqrt{dT_{\max}}}$ yields :

$$\begin{aligned}&\mathbb{E}\left[\|\nabla f(w_T)\|^2\right] \\ &\leq C_1\sqrt{\frac{d}{T_{\max}}} + C_2\frac{1}{T_{\max}} \\ &+ D_1\frac{\eta}{T_{\max}}\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\left\|\hat{v}_{t-1}^{-1/2}m_t\right\|^2\right] + D_2\frac{\eta}{T_{\max}}\sum_{t=1}^{T_{\max}}\mathbb{E}\left[\left\|\hat{v}_{t-1}^{-1/2}\tilde{g}_t\right\|^2\right]\end{aligned} \tag{45}$$

where

$$\begin{aligned}C_1 &= \frac{\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\Delta f + \frac{4L\left(\frac{\beta_1}{1-\beta_1}\right)^2\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\frac{(1+\beta_1^2)(1-\beta_1)}{(1-\beta_2)(1-\gamma)} \\ C_2 &= \frac{\mathsf{M}}{(1-\beta_1)\left((1-a\beta_1)+(\beta_1+a)\right)}\tilde{\mathsf{M}}^2\mathbb{E}\left[\left\|\hat{v}_0^{-1/2}\right\|\right]\end{aligned} \tag{46}$$

Simple case as in [31]: if $\beta_1 = 0$ then $\tilde{g}_t = g_t + m_{t+1}$ and $g_t = \theta_t$. Also using Lemma 3 we have that:

$$\sum_{t=1}^{T_{\max}}\eta_t^2\mathbb{E}\left[\left\|\hat{v}_t^{-1/2}g_t\right\|_2^2\right] \leq \frac{\eta^2 d T_{\max}}{(1-\beta_2)} \tag{47}$$

which leads to the final bound:

$$\begin{aligned}&\mathbb{E}\left[\|\nabla f(w_T)\|^2\right]\\&\leq \tilde{C}_1\sqrt{\frac{d}{T_{\max}}}+\tilde{C}_2\frac{1}{T_{\max}}\end{aligned} \tag{48}$$

where

$$\begin{aligned}\tilde{C}_1 &= C_1 + \frac{\mathsf{M}}{(1-a\beta_1)+(\beta_1+a)}\left[\frac{a(1-\beta_1)^2}{1-\beta_2}+2L\frac{1}{1-\beta_2}\right]\\ \tilde{C}_2 &= C_2 = \frac{\mathsf{M}}{(1-\beta_1)\left((1-a\beta_1)+(\beta_1+a)\right)}\tilde{\mathsf{M}}^2\mathbb{E}\left[\left\|\hat{v}_0^{-1/2}\right\|\right]\end{aligned} \tag{49}$$

□

C Proof of Lemma 4 (Boundedness of the iterates)

Lemma. *Given the multilayer model (12), assume the boundedness of the input data and of the loss function,* i.e., *for any $\xi \in \mathbb{R}^l$ and $y \in \mathbb{R}$ there is a constant $T > 0$ such that:*

$$\|\xi\| \leq 1 \quad a.s. \quad and |\mathcal{L}'(\cdot, y)| \leq T \tag{50}$$

where $\mathcal{L}'(\cdot, y)$ denotes its derivative w.r.t. *the parameter. Then for each layer $\ell \in [1, L]$, there exist a constant $A_{(\ell)}$ such that:*

$$\left\|w^{(\ell)}\right\| \leq A_{(\ell)}$$

Proof Recall that for any layer index $\ell \in [1, L]$ we denote the output of layer ℓ by $h^{(\ell)}(w, \xi)$:

$$h^{(\ell)}(w,\xi) = \sigma\left(w^{(\ell)}\sigma\left(w^{(\ell-1)}\ldots\sigma\left(w^{(1)}\xi\right)\right)\right)$$

Given the sigmoid assumption we have $\left\|h^{(\ell)}(w,\xi)\right\| \leq 1$ for any $\ell \in [1, L]$ and any $(w, \xi) \in \mathbb{R}^d \times \mathbb{R}^l$. Observe that at the last layer L:

$$\begin{aligned}\|\nabla_{w^{(L)}}\mathcal{L}(\mathsf{MLN}(w,\xi),y)\| &= \|\mathcal{L}'(\mathsf{MLN}(w,\xi),y)\nabla_{w^{(L)}}\mathsf{MLN}(w,\xi)\|\\ &= \left\|\mathcal{L}'(\mathsf{MLN}(w,\xi),y)\sigma'(w^{(L)}h^{(L-1)}(w,\xi))h^{(L-1)}(w,\xi)\right\|\\ &\leq \frac{T}{4}\end{aligned} \tag{51}$$

where the last equality is due to mild assumptions (50) and to the fact that the norm of the derivative of the sigmoid function is upperbounded by $1/4$.

From Algorithm 2, with $\beta_1 = 0$ we have for iteration index $t > 0$:

$$\begin{aligned}\|w_t - w_{t-1}\| &= \left\|-\eta_t\hat{v}_t^{-1/2}(\theta_t + h_{t+1})\right\|\\ &= \left\|\eta_t\hat{v}_t^{-1/2}(g_t + m_{t+1})\right\|\\ &\leq \hat{\eta}\left\|\hat{v}_t^{-1/2}g_t\right\| + \hat{\eta}a\left\|\hat{v}_t^{-1/2}g_{t+1}\right\|\end{aligned} \tag{52}$$

where $\hat{\eta} = \max_{t>0}\eta_t$. For any dimension $p \in [1, d]$, using assumption H 4, we note that

$$\sqrt{\hat{v}_{t,p}} \geq \sqrt{1-\beta_2}g_{t,p} \quad \text{and} \quad m_{t+1} \leq a\left\|g_{t+1}\right\|$$

. Thus:

$$\begin{aligned}\|w_t - w_{t-1}\| &\leq \hat{\eta}\left(\left\|\hat{v}_t^{-1/2}g_t\right\| + a\left\|\hat{v}_t^{-1/2}g_{t+1}\right\|\right)\\ &\leq \hat{\eta}\frac{a+1}{\sqrt{1-\beta_2}}\end{aligned} \tag{53}$$

In short there exist a constant B such that $\|w_t - w_{t-1}\| \leq B$.

Proof by induction: As in [6], we will prove the containment of the weights by induction. Suppose an iteration index T and a coordinate i of the last layer L such that $w_{T,i}^{(L)} \geq \frac{T}{4\lambda} + B$. Using (51), we have

$$\nabla_i f(w_t^{(L)} \geq -\frac{T}{4} + \lambda \frac{T}{\lambda 4} \geq 0$$

where $f(\cdot)$ is defined by (13) and is the loss of our MLN. This last equation yields $\theta_{T,i}^{(L)} \geq 0$ (given the algorithm and $\beta_1 = 0$) and using the fact that $\|w_t - w_{t-1}\| \leq B$ we have

$$0 \leq w_{T-1,i}^{(L)} - B \leq w_{T,i}^{(L)} \leq w_{T-1,i}^{(L)} \tag{54}$$

which means that $|w_{T,i}^{(L)}| \leq w_{T-1,i}^{(L)}$. So if the first assumption of that induction reasoning holds, *i.e.*, $w_{T-1,i}^{(L)} \geq \frac{T}{4\lambda} + B$, then the next iterates $w_{T,i}^{(L)}$ decreases, see (54) and go below $\frac{T}{4\lambda} + B$. This yields that for any iteration index $t > 0$ we have

$$w_{T,i}^{(L)} \leq \frac{T}{4\lambda} + 2B$$

since B is the biggest jump an iterate can do since $\|w_t - w_{t-1}\| \leq B$. Likewise we can end up showing that

$$|w_{T,i}^{(L)}| \leq \frac{T}{4\lambda} + 2B$$

meaning that the weights of the last layer at any iteration is bounded in some matrix norm.

Now that we have shown this boundedness property for the last layer L, we will do the same for the previous layers and conclude the verification of assumption H 2 by induction.

For any layer $\ell \in [1, L-1]$, we have:

$$\nabla_{w^{(\ell)}} \mathcal{L}(\mathsf{MLN}(w,\xi), y) = \mathcal{L}'(\mathsf{MLN}(w,\xi), y) \left(\prod_{j=1}^{\ell+1} \sigma' \left(w^{(j)} h^{(j-1)}(w,\xi) \right) \right) h^{(\ell-1)}(w,\xi) \tag{55}$$

This last quantity is bounded as long as we can prove that for any layer ℓ the weights $w^{(\ell)}$ are bounded in some matrix norm as $\left\|w^{(\ell)}\right\|_F \leq F_\ell$ with the Frobenius norm. Suppose we have shown $\left\|w^{(r)}\right\|_F \leq F_r$ for any layer $r > \ell$. Then having this gradient (55) bounded we can use the same lines of proof for the last layer L and show that the norm of the weights at the selected layer ℓ satisfy

$$\left\| w^{(\ell)} \right\| \leq \frac{T \prod_{t > \ell} F_t}{4^{L-\ell+1}} + 2B$$

Showing that the weights of the previous layers $\ell \in [1, L-1]$ as well as for the last layer L of our fully connected feed forward neural network are bounded at each iteration, leads by induction, to the boundedness (at each iteration) assumption we want to check. □