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 Reddi et al. [2018] 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 some 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 bound under some conditions. We conduct experiments on several deep learning models, which show that by improving sample efficiency, the proposed method achieves a convergence speedup in practice.

1 Introduction

Nowadays deep learning has been very successful in several applications, from robotics (e.g. Levine et al. [2017]), computer vision (e.g. He et al. [2016], Goodfellow et al. [2014]), reinforcement learning (e.g. Mnih et al. [2013]), to natural language processing (e.g. Graves et al. [2013]). A common goal in these applications is learning quickly. It becomes a desired goal due to the presence of big data and/or the use of large neural nets. To accelerate the process, there are number of algorithms proposed in recent years, such as AMSGRAD Reddi et al. [2018], ADAM Kingma and Ba [2015], RMSPROP Tieleman and Hinton [2012], ADADELTA Zeiler [2012], and NADAM Dozat [2016], etc.

All the prevalent algorithms for training deep nets mentioned above combine two ideas: the idea of adaptivity from ADAGRAD Duchi et al. [2011], McMahan and Streeter [2010] and the idea of momentum from NESTEROV'S METHOD Nesterov [2004] or HEAVY BALL method Polyak [1964]. 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: the learning rate is different for each dimension, depending on the magnitude of gradient in each dimension, which might help in exploiting the geometry of data and leading to a better update. On the other hand, NESTEROV'S METHOD or HEAVY BALL Method Polyak [1964] 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 Reddi et al. [2018] and ADAM Kingma and Ba [2015] leverage this ideas to accelerate the training process of neural nets.

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 Chiang et al. [2012], Rakhlin and Sridharan [2013], Syrgkanis et al. [2015], Abernethy et al. [2018], 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 can 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 — OPTIMISTIC-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 theoretical analysis of OPTIMISTIC-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 in online learning, as we use some tools from it to design and analyze the proposed algorithm. We follow the notations in related adaptive optimization papers Kingma and Ba [2015], Reddi et al. [2018]. 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 $w_t \in \mathcal{K} \subseteq \mathbb{R}^d$, then the learner observes $\ell_t(\cdot)$ and suffers loss $\ell_t(w_t)$ after the learner commits the action. The goal of the learner is minimizing 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 $w^* \in \mathcal{K}$.

The idea of OPTIMISTIC ONLINE LEARNING (e.g. Chiang et al. [2012], Rakhlin and Sridharan [2013], Syrgkanis et al. [2015], Abernethy et al. [2018]) is as follows. Suppose that, in each round t, the learner has a good guess $m_t(\cdot)$ of the loss function $\ell_t(\cdot)$ before playing an action w_t. Then, the learner should exploit the guess $m_t(\cdot)$ to choose an action w_t since $m_t(\cdot)$ is close to the true loss function $\ell_t(\cdot)$. [1] For example, Syrgkanis et al. [2015] proposes an optimistic-variant of FOLLOW-THE-REGULARIZED-LEADER (FTRL). FTRL (see e.g. Hazan [2016]) is an online learning algorithm whose update is

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

where η is a parameter, $R(\cdot)$ is a 1-strongly convex function with respect to a norm ($\|\cdot\|$) on the constraint set $\mathcal{K}$, and $L_{t-1} := \sum_{s=1}^{t-1} g_s$ is the cumulative sum of gradient vectors of the loss functions (i.e. $g_s := \nabla \ell_s(w_s)$) up to but not including t. FTRL has regret at most $O(\sqrt{\sum_{t=1}^{T} \|g_t\|_*})$. On the other hand, OPTIMISTIC-FTRL Syrgkanis et al. [2015] has update

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

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 would be only a constant factor worse than that of its counterpart FTRL. In Section **??**, we will provide a way to get m_t. Now we just want to emphasize 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 OPTIMISTIC-AMSGRAD and AMSGRAD.

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

2.2 Adaptive optimization methods

Recently, adaptive optimization has been popular in various deep learning applications due to their superior empirical performance. ADAM Kingma and Ba [2015] is a very popular adaptive algorithm for training deep nets. It combines the momentum idea Polyak [1964] with the idea of ADAGRAD Duchi et al. [2011], 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. This adaptive learning rate might help for accelerating the convergence when the gradient vector is sparse Duchi et al. [2011]. However, when applying ADAGRAD to train deep nets, it is observed that the learning rate might decay too fast Kingma and Ba [2015]. Therefore, Kingma and Ba [2015] proposes using 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 (i.e. line 5,6 and line 8 of Algorithm 1). Yet, ADAM Kingma and Ba [2015] fails at some online convex optimization problems. AMSGRAD Reddi et al. [2018] fixes the issue. The algorithm of AMSGRAD is shown in Algorithm 1. The difference between ADAM and AMSGRAD lies on 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 Reddi et al. [2018] 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 $\frac{\text{Regret}_T}{T} \rightarrow 0$). For the hyper-parameters of AMSGRAD, it is suggested in Reddi et al. [2018] that $\beta_1 = 0.9$ and $\beta_2 = 0.99$.

Algorithm 1 AMSGRAD Reddi et al. [2018]

```
1: Required: parameter $\beta_1$, $\beta_2$, and $\eta_t$.
2: Init: $w_1 \in \mathcal{K} \subseteq \mathbb{R}^d$ and $v_0 = \epsilon 1 \in \mathbb{R}^d$.
3: for $t = 1$ to $T$ do
4:     Get mini-batch stochastic gradient vector $g_t$ 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 OPTIMISTIC-AMSGRAD

We propose a new optimization algorithm, OPTIMISTIC-AMSGRAD, shown in Algorithm 2. It combines the idea of adaptive optimization with optimistic learning. In 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 $w_{t+\frac{1}{2}} \in \mathcal{K}$ (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}. As the learner's action set is $\mathcal{K} \subseteq \mathbb{R}^d$, we adopt the notation $\Pi_{\mathcal{K}}[\cdot]$ for the projection to $\mathcal{K}$ if needed. The scheme of AMSGRAD is summarized in Figure 1.

Algorithm 2 OPTIMISTIC-AMSGRAD

```
1: Input: Parameters $\beta_1, \beta_2, \epsilon, \eta_k$
2: Init.: $w_1 = w_{-1/2} \in \mathcal{K} \subseteq \mathbb{R}^d$ and $v_0 = \epsilon \mathbf{1} \in \mathbb{R}^d$
3: for $k = 1, 2, \dots, K$ do
4:     Get mini-batch stochastic gradient $g_k$ at $w_k$
5:     $\theta_k = \beta_1 \theta_{k-1} + (1-\beta_1)\, g_k$
6:     $v_k = \beta_2 v_{k-1} + (1-\beta_2)\, g_k^2$
7:     $\hat{v}_k = \max(\hat{v}_{k-1}, v_k)$
8:     $w_{k+\frac{1}{2}} = \Pi_{\mathcal{K}}\left[w_k - \eta_k \frac{\theta_k}{\sqrt{\hat{v}_k}}\right]$
9:     $w_{k+1} = \Pi_{\mathcal{K}}\left[w_{k+\frac{1}{2}} - \eta_k \frac{h_{k+1}}{\sqrt{\hat{v}_k}}\right]$ where    $h_{k+1} := \beta_1 \theta_{k-1} + (1-\beta_1) m_{k+1}$
10:        and    $m_{k+1}$ is a guess of $g_{k+1}$
11: end for
12: Return: $w_{K+1}$.
```

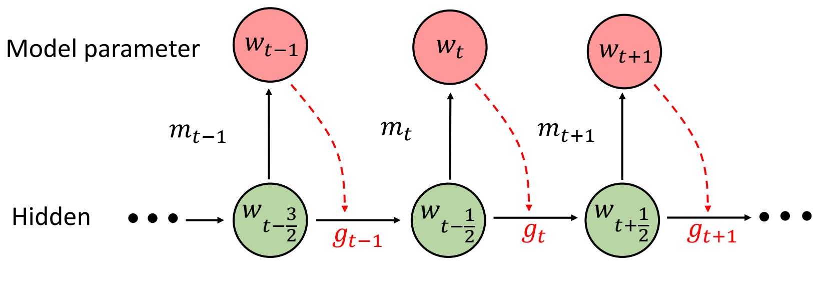

Figure 1: Scheme of OPTIMISTIC-AMSGRAD.

We see that the proposed OPTIMISTIC-AMSGRAD inherits three properties:

- Adaptive learning rate of each dimension as ADAGRAD Duchi et al. [2011]. (line 6, line 8 and line 9)
- Exponential moving average of the past gradients as NESTEROV'S METHOD Nesterov [2004] and the HEAVY-BALL method Polyak [1964]. (line 5)
- Optimistic update that exploits a good guess of the next gradient vector as optimistic online learning algorithms Chiang et al. [2012], Rakhlin and Sridharan [2013], Syrgkanis et al. [2015]. (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 OPTIMISTIC-AMSGRAD.

Observe that the proposed algorithm does not reduce to AMSGRAD when $m_t = 0$. Furthermore, if $\mathcal{K} = \mathbb{R}^d$ (unconstrained case), one might want to combine line 8 and line 9 and get a single line as

$$w_{k+1} = w_k - \eta_k \frac{\theta_k}{\sqrt{\hat{v}_k}} - \eta_k \frac{h_{k+1}}{\sqrt{\hat{v}_k}} \tag{3}$$

Yet, based on this expression, we see that w_{t+1} is updated from $w_{t-\frac{1}{2}}$ instead of w_t. Therefore, while OPTIMISTIC-AMSGRAD looks like just doing an additional update compared to AMSGRAD, the difference of the updates is subtle. In the following analysis, we show that the interleaving actually leads to some cancellation in the regret bound.

3.1 Theoretical analysis

TO COMPLETE

In this section, we provide regret analysis of the proposed method and show that it may improve the bound of vanilla AMSGRAD with good guess of gradient.

Notations. To begin with, let us introduce some notations first. We denote the Mahalanobis norm $\| \cdot \|_H := \sqrt{\langle \cdot, H \cdot \rangle}$ for some PSD matrix H. We let $\psi_t(x) := \langle x, \text{diag}\{\hat{v}_t\}^{1/2} x \rangle$ for a PSD matrix $H_t^{1/2} := \text{diag}\{\hat{v}_t\}^{1/2}$, where $\text{diag}\{\hat{v}_t\}$ represents the diagonal matrix whose i_{th} diagonal element is $\hat{v}_t[i]$ in Algorithm 2. We define its corresponding Mahalanobis norm $\| \cdot \|_{\psi_t} := \sqrt{\langle \cdot, \text{diag}\{\hat{v}_t\}^{1/2} \cdot \rangle}$, where we abuse the notation ψ_t to represent the PSD matrix $H_t^{1/2} := \text{diag}\{\hat{v}_t\}^{1/2}$. Consequently, $\psi_t(\cdot)$ is 1-strongly convex with respect to the norm $\| \cdot \|_{\psi_t} := \sqrt{\langle \cdot, \text{diag}\{\hat{v}_t\}^{1/2} \cdot \rangle}$. Namely, $\psi_t(\cdot)$ satisfies $\psi_t(u) \geq \psi_t(v) + \langle \psi_t(v), u - v \rangle + \frac{1}{2} \|u - v\|_{\psi_t}^2$ for any point u, v. A consequence of 1-strongly convexity of $\psi_t(\cdot)$ is that $B_{\psi_t}(u, v) \geq \frac{1}{2} \|u - v\|_{\psi_t}^2$, where the Bregman divergence $B_{\psi_t}(u, v)$ is defined as $B_{\psi_t}(u, v) := \psi_t(u) - \psi_t(v) - \langle \psi_t(v), u - v \rangle$ with $\psi_t(\cdot)$ as the distance generating function. We can also define the corresponding dual norm $\| \cdot \|_{\psi_t^*} := \sqrt{\langle \cdot, \text{diag}\{\hat{v}_t\}^{-1/2} \cdot \rangle}$.

Non-asymptotic analysis. We establish in this section a finite-time upper bound of the expected squared norm of the gradient of the objective function we are optimizing. The objective function for most deep learning task reads as follows:

$$\min_{w \in \Theta} f(w) := \frac{1}{n} \sum_{i=1}^{n} f(w, \xi_i) \tag{4}$$

where $(\xi_i, i \in [1, n])$ is the vector of n input data.

3.2 Comparison to some related methods

Comparison to nonconvex optimization works. Recently, Zaheer et al. [2018], Chen et al. [2019a], Ward et al. [2019], Zhou et al. [2018], Zou and Shen [2018], Li and Orabona. [2019] provide some theoretical analysis of ADAM-type algorithms when applying them to smooth nonconvex optimization problems. For example, Chen et al. [2019a] provides a bound, which is $\min_{t \in [T]} \mathbb{E}[\|\nabla f(w_t)\|^2] = O(\log T / \sqrt{T})$. Yet, this data independent bound does not show any advantage over standard stochastic gradient descent. Similar concerns appear in other papers.

To get some adaptive data dependent bound that are in terms of the gradient norms observed along the trajectory) when applying OPTIMISTIC-AMSGRAD to nonconvex optimization, one can follow the approach of Agarwal et al. [2019] or Chen et al. [2019b]. They provide ways to convert algorithms with adaptive data dependent regret bound for convex loss functions (e.g. ADAGRAD) to the ones that can find an approximate stationary point of non-convex loss functions. Their approaches are modular so that simply using OPTIMISTIC-AMSGRAD as the base algorithm in their methods will immediately lead to a variant of OPTIMISTIC-AMSGRAD that enjoys some guarantee on nonconvex optimization. The variant can outperform the ones instantiated by other ADAM-type algorithms when the gradient prediction m_t is close to g_t. The details are omitted since this is a straightforward application.

Comparison to AO-FTRL Mohri and Yang [2016]. In Mohri and Yang [2016], the authors propose AO-FTRL, which has the update of the form $w_{t+1} = \arg\min_{w \in \mathcal{K}} (\sum_{s=1}^{t} g_s)^\top w + m_{t+1}^\top w + r_{0:t}(w)$, where $r_{0:t}(\cdot)$ is a 1-strongly convex loss function with respect to some norm $\| \cdot \|_{(t)}$ that may be different for different iteration t. Data dependent regret bound was provided in the paper, which is $r_{0:T}(w^*) + \sum_{t=1}^{T} \|g_t - m_t\|_{(t)^*}$ for any benchmark $w^* \in \mathcal{K}$. We see that if one selects $r_{0:t}(w) := \langle w, \text{diag}\{\hat{v}_t\}^{1/2} w \rangle$ and $\| \cdot \|_{(t)} := \sqrt{\langle \cdot, \text{diag}\{\hat{v}_t\}^{1/2} \cdot \rangle}$, then the update might be viewed as an optimistic variant of ADAGRAD. However, no experiments was provided in Mohri and Yang [2016].

Comparison to OPTIMISTIC-ADAM Daskalakis et al. [2018]. We are aware that Daskalakis et al. [2018] proposed one version of optimistic algorithm for ADAM, which is called OPTIMISTIC-ADAM in their paper. A slightly modified version is summarized in Algorithm 3. Here, OPTIMISTIC-ADAM+$\hat{v}_t$ is OPTIMISTIC-ADAM in Daskalakis et al. [2018] with the additional max operation $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$ to guarantee that the weighted second moment is monotone increasing.

Algorithm 3 OPTIMISTIC-ADAM DASKALAKIS ET AL. [2018]+$\hat{v}_t$.

```
1: Required: parameter $\beta_1$, $\beta_2$, and $\eta_t$.
2: Init: $w_1 \in \mathcal{K}$ and $\hat{v}_0 = v_0 = \epsilon 1 \in \mathbb{R}^d$.
3: for $t = 1$ to $T$ do
4:     Get mini-batch stochastic gradient vector $g_t \in \mathbb{R}^d$ 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:     $w_{t+1} = \Pi_k [w_t - 2\eta_t \frac{\theta_t}{\sqrt{\hat{v}_t}} + \eta_t \frac{\theta_{t-1}}{\sqrt{\hat{v}_{t-1}}}]$.
9: end for
```

We want to emphasize that the motivations are different. OPTIMISTIC-ADAM in their paper is designed to optimize two-player games (e.g. GANs Goodfellow et al. [2014]), while the proposed algorithm in this paper is designed to accelerate optimization (e.g. solving empirical risk minimization quickly). Daskalakis et al. [2018] focuses on training GANs Goodfellow et al. [2014]. GANs is a two-player zero-sum game. There have been some related works in OPTIMISTIC ONLINE LEARNING like Chiang et al. [2012], Rakhlin and Sridharan [2013], Syrgkanis et al. [2015]) showing that if both players use some kinds of OPTIMISTIC-update, then accelerating the convergence to the equilibrium of the game is possible. Daskalakis et al. [2018] was inspired by these related works and showed that OPTIMISTIC-MIRROR-DESCENT can avoid the cycle behavior in a bilinear zero-sum game, which accelerates the convergence. Furthermore, Daskalakis et al. [2018] did not provide theoretical analysis of OPTIMISTIC-ADAM.

3.3 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.

References

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

N. Agarwal, B. Bullins, X. Chen, E. Hazan, K. Singh, C. Zhang, and Y. Zhang. Efficient full-matrix adaptive regularization. *ICML*, 2019.

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

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

X. Chen, S. Liu, R. Sun, and M. Hong. On the convergence of a class of adam-type algorithms for non-convex optimization. *ICLR*, 2019a.

Z. Chen, Z. Yuan, J. Yi, B. Zhou, E. Chen, and T. Yang. Universal stagewise learning for non-convex problems with convergence on averaged solutions. *ICLR*, 2019b.

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.

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

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

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

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

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

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

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

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

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

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.

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

X. Li and F. Orabona. On the convergence of stochastic gradient descent with adaptive stepsizes. *AISTAT*, 2019.

J. Martens and R. Grosse. Optimizing neural networks with kronecker-factored approximate curvature. *ICML*, 2015.

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

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.

M. Mohri and S. Yang. Accelerating optimization via adaptive prediction. *AISTATS*, 2016.

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

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

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

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

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

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

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

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

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

R. Ward, X. Wu, and L. Bottou. Adagrad stepsizes: Sharp convergence over nonconvex landscapes, from any initialization. *ICML*, 2019.

M. Zaheer, S. Reddi, D. Sachan, S. Kale, and S. Kumar. Adaptive methods for nonconvex optimization. *NeurIPS*, 2018.

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

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

F. Zou and L. Shen. On the convergence of adagrad with momentum for training deep neural networks. *arXiv:1808.03408*, 2018.

A Proof of Theorem ??

Proof □