A NEW VARIANT OF STOCHASTIC HEAVY BALL OPTIMIZATION METHOD FOR DEEP LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Stochastic momentum optimization methods, also known as stochastic heavy ball (SHB) methods, are one of the most popular optimization methods for deep learning. These methods can help accelerate stochastic gradient descent and dampen oscillations. In this paper we provide a new variant of the stochastic heavy ball method, called stochastic Euler's heavy ball (SEHB). The proposed SEHB method modifies the steepest descent direction to achieve acceleration, and combines Euler's method to adaptively adjust learning rates as well. A convergence analysis of the regret bound is discussed under the online convex optimization framework. Furthermore, we conduct experiments on various popular datasets and deep learning models. Empirical results demonstrate that our SEHB method shows comparable or even better generalization performance than state-of-the-art optimization methods such as SGD and Adam.

1 INTRODUCTION

Stochastic gradient descent (SGD) is the main optimization method for deep learning. This method is often trained in the form of mini-batch SGD in order to meet the requirements of computing power, achieving great success in various machine learning and deep learning tasks (Krizhevsky et al., 2012; Graves et al., 2013; Lecun et al., 1998). However, SGD havs two main drawbacks: one is the use of the negative gradient of loss functions as descent directions which leads to a slow convergence near the local minima and is difficult to escape from local suboptimal solutions; the other one is that SGD scales the gradient uniformly in all directions which may lead to poor performance as well as limited training speed. In recent years, considerable efforts have been spent on improving SGD and several remarkable variants have been proposed.

The two main ways for improving SGD are adjusting the steepest descent directions and applying adaptive learning rates to the training process, called stochastic momentum methods and adaptive methods respectively. Stochastic momentum methods, including stochastic heavy ball (SHB) (Polyak, 1964) and stochastic Nesterov's accelerated gradient (SNAG) (Nesterov, 1983), have a smoother convergence process and generalize comparable or even better than SGD; Adaptive methods enjoy rapid training speed and need less effort to adjust the hyperparameters like initial learning rates, among which the representative algorithms are AdaGrad (Duchi et al., 2011), AdaDelta (Zeiler, 2012) and RMSprop (Tieleman & Hinton, 2012).

Adam (Kingma & Ba, 2015), as one of the most popular algorithms for deep learning, is a combination of stochastic momentum methods with adaptive methods. Adam has the most adavantages of both methods but it seems to have poorer generalization ability and out-sample behavior than SGD and stochastic momentum methods. Recently, research shows that Adam even fails to converge on some convex functions (Reddi et al., 2018). Many variants of Adam have been proposed to solve this issue, such as AMSGrad (Reddi et al., 2018) and AdaBound (Luo et al., 2019). However, the generalization ability of AMSGrad on unseen data is found to be similar to that of Adam and there exists a considerable performance gap between AMSGrad and SGD or SHB (Keskar & Socher, 2017).

A new optimization method, called Adaptive Energy Gradient Descent (AEGD) (Tan, 2020), has been proposed and demonstrates a novel direction to improve the generalization ability of SGD. AEGD finds the connection between numerical methods ,which are used to solve partial differential equations (PDEs), and SGD by successfully applying Invariant Energy Quadratization(IEQ) (Yang,

2016) to SGD. IEQ is introduced for gradient flows in the form of PDE and constructs linear, unconditionally energy stable schemes for solving time-dependent PDEs.

In this paper, we first point out that the iteration rule used in IEQ and AEGD is similar to Euler's method, which is a classical method for analyzing differential equations. When we use it in SGD, it's equivalent to adaptively adjust the learning rates during the descent by multipling a vector element-wisely. Inspired by the aforementioned analysis, we propose a new variant of SHB, called stochastic Euler's heavy ball (SEHB), by applying Euler's method to SHB. It's a combination of adaptive learning rate adjustments with SHB. Moreover, we give a convergence analysis of SEHB by proving the regret bound under online convex optimization framework (Zinkevich, 2003). Finally, we conduct several representative experiments in order to test the performance of SEHB and other classical optimization methods for deep learning. There are four main experiments including numerical experiment on Extended Rosenbrock function and three neural network experiments: fully connected neural networks on MNSIT datasets, ResNet-50 on CIFAR-10 dataset and DenseNet-121 on CIFAR-10 dataset.

2 Notation and Preliminaries

Notation. For a vector $\theta \in \mathbb{R}^d$, we denote its i-th coordinate by θ_i; we use θ_t to denote θ in the t-th iteration and use $\theta_{t,i}$ for the i-th coordinate of θ in the t-th iteration. Furthermore, we use $||\cdot||$ to denote l_2-norm and use $||\cdot||_\infty$ to denote l_∞-norm. Given two vectors $v, w \in \mathbb{R}^d$, we use vw to denote element-wise product and use v^2 to denote element-wise square; we use $\frac{1}{v}$ to denote element-wise division.

Online convex optimization framework. We analyze the convergence of our method SEHB under the online learning framework which is one of flexible frameworks to analyze iterative optimization methods. In this online setup, at each step t, the optimization algorithm chooses a decision $x_t \in \mathcal{F}$ where $\mathcal{F} \in \mathbb{R}^d$ is a convex feasible set. Then the decision θ_t is evaluated by a convex loss function f_t. The optimization algorithm's regret at the end of T steps is given by $R(T) = \sum_{t=1}^{T} f_t(\theta_t) - \min_{\theta \in \mathcal{F}} \sum_{t=1}^{T} f_t(\theta)$. The regret evaluates the average performance of the algorithm and a $o(T)$ vanishing regret implies the algorithm's performance converges to the optimal one on average.

Euler's method Euler's method is a technique used to analyze a differential equation. Motivated by the concept of local linearity or linear approximation, the method uses small tangent lines over a short distance to approximate the solution of the initial value problem. Let $x \in \mathbb{R}^d, f(x) : \mathbb{R}^d \to \mathbb{R}$ and we use r to approximate f, the update rule is $r_{t+1} = r_t + \nabla f(x_t)(x_{t+1} - x_t)$. Thus, if the sequence interval of $\{x_t\}$ approaches to 0, r is a good fitting of f.

3 Algorithm and Convergence Analysis

In this section, we develop a new variant of the stochastic heavy ball method, which we call stochastic Euler's heavy ball (SEHB), and provide the convergence analysis of SEHB under online convex optimization framework.

3.1 Algorithm

We aim to devise a new strategy that combine the rapid initial progress of adaptive methods and the good final generalization properties of SGD and SHB. Algorithm 1 is the pseudo code of SEHB where all operations are element-wise.

To understand the design of our algorithm, we first let $g(\theta_t) = \sqrt{f(\theta_t) + c}$ where $f(\theta_t)$ is the loss function of models and c is a constant number which guarantees $f(\theta_t) + c > 0$. Thus we have $\nabla f(\theta_t) = 2g(\theta_t)\nabla g(\theta_t)$. Inspired by AEGD and IEQ, we use Euler's method to approximate the function $g(\theta_t)$

$$r_{t+1} = r_t + \nabla g(\theta_t)(\theta_{t+1} - \theta_t), \tag{1}$$

Algorithm 1 SEHB

```
Input: $\theta_1 \in \mathbb{R}^d$, step size $\{\eta_t\}_{t=1}^T$, $c = 1$, $\{\gamma_t\}_{t=1}^T$
1: Set $v_{0,i} = 0$, $r_{0,i} = \sqrt{f(\theta_1) + c}$, $i = 1, 2, \cdots, d$
2: for $t = 1$ to $T$ do
3:     $G \leftarrow \nabla f(\theta_t)/2\sqrt{f(\theta_t) + c}$      (compute the gradient of $g$)
4:     $r_{t+1} \leftarrow r_t/(1 + 2\eta_t G^2)$      (update vector $r$ with Euler's method)
5:     $v_{t+1} \leftarrow \gamma_t v_t + G$      (update SHB direction of $g$)
6:     $\theta_{t+1} \leftarrow \theta_t - 2\eta_t r_{t+1} v_{t+1}$      (update $\theta$)
7: end for
```

where the inital value $r_{0,i} = g(\theta_1)$, $i = 1, 2, \cdots, d$ and the SHB descent direction is given by

$$v_{t+1} = \gamma_t v_t + \nabla g(\theta_t). \tag{2}$$

Because the iteration rule

$$\theta_{t+1} = \theta_t - 2\eta_t r_{t+1} v_{t+1} \tag{3}$$

includes r_{t+1} which relies on θ_{t+1} to update, we need to remove θ_{t+1} from the update rule of r_{t+1}. Let the SHB decay parameter $\gamma_t = 0$, and plugging θ_{t+1} into the update rule of r_{t+1} leads to

$$r_{t+1} = r_t/(1 + 2\eta_t(\nabla g(\theta_t))^2). \tag{4}$$

We use the above rule to update r_{t+1} in practice. Now let us revisit the update rule of θ_{t+1}, it is equivalent to

$$\theta_{t+1} = \theta_t - \eta_t \frac{r_{t+1}}{\sqrt{f(\theta_t) + c}} \nabla f(\theta_t). \tag{5}$$

It seems that the only difference from SGD is the learning rate which has an adaptive adjustment by multipling $r_{t+1}/\sqrt{f(\theta_t) + c}$. Therefore, by letting the SHB decay parameter $\gamma_t \in (0, 1)$, SEHB realizes the combination of SHB and the adaptive adjustment of learning rates.

Considering the fact that SEHB use SHB descent direction in the iteration process, we still use 0.9 as the default of γ_t in practice for any time step t but when analyzing the convergence of SEHB, the default setting is $\gamma_1 \lambda^{t-1}$ under the online convex optimization framework, which is as same as β_{1t} in Adam and AMSGrad. Given the hyperparameter c, Equation (5) implies that $1/\sqrt{f(\theta_t) + c}$ participates in adjusting learning rates. If it is very close to 0 at a certain step of the iteration process, learning rates would become pathological and the iterative process would become unstable. Therefore, to avoid this issue, we set the default of c to 1 based on the fact that the most of loss functions for deep learning have a lower bound 0. Finally, we choose 0.01 as the default learning rate of SEHB which comes from empirical results. However, it may be not the best learning rate in a certain experiment but can show a relatively excellent performance if users don't want to spend any effort on tuning.

3.2 Convergence Analyze

We prove the convergence of SEHB under the online convex optimization framework. The key results are as follows:

Theorem 1. Suppose that the fucntion g_t has bounded gradients, $||\nabla g_t(\theta)||_\infty \leq G_\infty$ for all $\theta \in \mathbb{R}^d$ and is bounded in feasible regions, $|g_t(\theta)| \leq g_\infty$. Suppose the distance between any θ_t generated by SEHB is bounded, $||\theta_n - \theta_m||_\infty \leq D_\infty$ for any $n, m \in \{1, 2, \cdots, T\}$, $T \geq 2$, and $\gamma_1 \in [0, 1)$. Let $\eta_t = \frac{\eta}{\sqrt{t}}$ and $\gamma_t = \gamma\lambda^{t-1}$, $\lambda \in (0, 1)$. Then SEHB has the following bound on the regret

$$R(T) \leq \frac{1}{2\eta} D_\infty^2 \sqrt{T} g_\infty \sum_{i=1}^d \left(\frac{1}{r_{T+1,i}} + \frac{1}{r_{2,i}}\right) + \frac{g_\infty D_\infty^2 d\gamma}{1-\lambda} + \frac{g_\infty d G_\infty^2 \gamma^2(\gamma + 2g_\infty^2 \lambda^2 \eta)}{\lambda^2(1-\gamma)^2(1-\lambda^2)}. \tag{6}$$

The following results falls as an immediate corollary of the above result:

Corollary 1. Under the same assumptions of Theorem 1, SEHB have the following average regret of convergence:

$$\frac{R(T)}{T} = O(\frac{1}{\sqrt{T}}) \tag{7}$$

Remark. The above bound in Equation (6) can be considered better than $O(\sqrt{dT})$ regret of SGD (Nesterov, 1983) when $\sum_{i=1}^{d}(\frac{1}{r_{T+1,i}} + \frac{1}{r_{2,i}}) \ll \sqrt{d}$. In our convergence analysis, decaying γ_t towards zero is important and several previous findings suggest that the SHB coefficient in the end of training can improve convergence (Sutskever et al., 2013). Finally, we find that the regret bound of SEHB is similar to the classical adaptive methods like Adam and AMSGrad. However, empirical results show that SEHB may generalize better than Adam and AMSGrad on several classic image classification tasks. We will show details in next section.

4 Experiemnt

In this section, we study the generalization performance of SEHB and several famous optimization methods including SHB (also known as SGDM), AdaGrad, Adam, AEGD, AMSGrad, AdaBound. We focus on four main experiments: numerical experiment on Extended Rosenbrock function; fully connected neural network experiment on MNIST dataset (Lecun et al., 1998); ResNet-50 (He et al., 2016) and DenseNet-121 (Huang et al., 2017) on CIFAR-10 dataset (Krizhevsky & Hinton, 2009).

4.1 Numerical Experiemnt

Extended Rosenbrock function is a famous high-dimensional non-convex function in optimization. Let $x \in \mathbb{R}^{1000}$, $f(x) : \mathbb{R}^{1000} \to \mathbb{R}$ where f is Extended Rosenbrock function. We start with initial point $x_0 = (-1.2, 1, -1.2, 1, \cdots, -1.2, 1)^T$ to find the optimal solution. The exact optimal solution is $(1, 1, \cdots, 1)^T$ and the exact minimum value of Extended Rosenbrock function is 0. We select three optimization methods to solve the optimization problem which include Adam, AEGD and SEHB. For hyperparameters in Adam, we use the default value of β_1 and β_2 and use $1e-2$ as the learning rate. We let $c = 1$ and choose $1e-4$ as the learning rate when using AEGD and SEHB. In addition, we use the default value 0.9 as γ_t in SEHB. Our criterion for choosing learning rates is to choose the one showing the fastest convergence with as little oscillation as possible. Figure 1 shows the result of the numerical experiment.

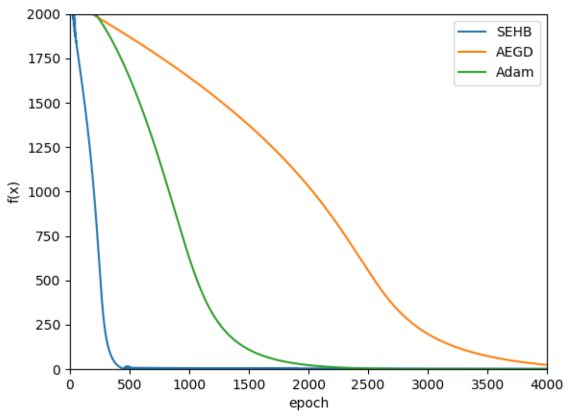

Figure 1: Performance comparison of Adam, AEGD and our method SEHB on Extended Rosenbrock function. All three methods converge to the optimal solution.

We note that although all three optimization methods have converged to the optimal solution, SEHB need the least epochs than Adam and AEGD. Under the requirement of the smooth iteration process, SEHB shows faster convergence than Adam and AEGD.

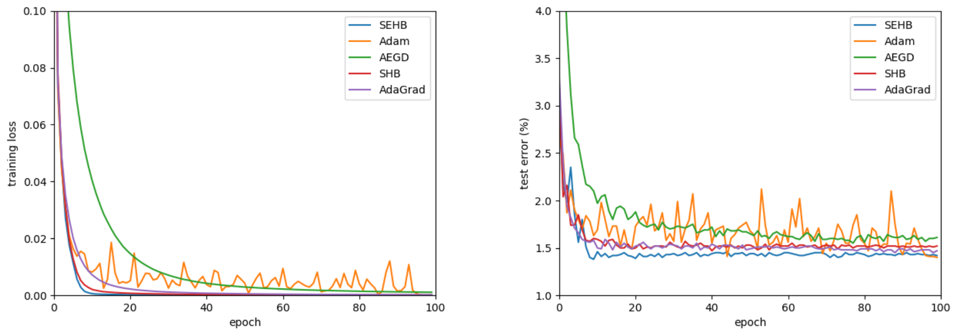

Figure 2: Performance comparison of fully connected neural networks with Adam, AEGD, AdaGrad, SHB and our method SEHB on MNIST dataset. The left is the training loss curve and the right is the test error curve.

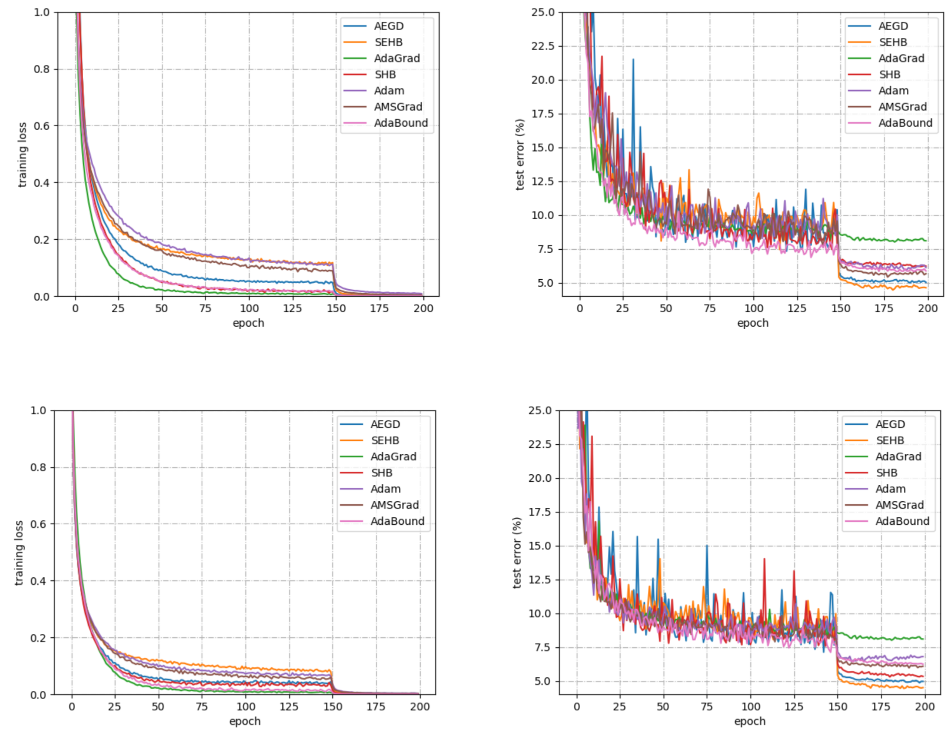

Figure 3: Performance comparison of deep convolution neural networks with Adam, AMSGrad, AdaBound, AEGD, AdaGrad, SHB and our method SEHB on CIFAR-10 dataset. The top row is the training loss curve and the test error curve of ResNet-50 while the bottom row is the training loss curve and the test error curve of DenseNet-121.

4.2 Neural Networks

In this subsection, we show the empirical results of several neural network models.

4.2.1 HYPERPARAMETERS TUNING

In the training of neural networks, hyperparameters settings have a significant impact on training results. Therefore, we need spend some efforts to tune hyperparameters of optimization methods for obtaining the best performance of each method. Here we show how to tune the hyperparameters. To tune the learning rate, we set five candidate learning rates and find the best one as the final learning rates to compare with other methods. For the hyperparameters of all methods, we tune them in the following ways:

SHB. The candidate learning rates of SHB are selected from $\{1, 0.5, 0.3, 0.1, 0.01\}$. We set γ, which is the coefficient of momentum in SHB, to the default value 0.9. Empirical results show that the default γ often have a good performance on most of the models. Otherwise, the learning rates vary on different models. For instance, the best learning rate of fully connected neural network on MNIST dataset and ResNet-50 on CIFAR-10 dataset is 0.1 while the best one of DenseNet-121 on CIFAR-10 dataset is 0.3.

AEGD. The candidate learning rates of AEGD are selected from $\{0.5, 0.3, 0.1, 0.05, 0.01\}$. We set c to 1. The best learning rate of AEGD is 0.1 for the three experiments.

AdaGrad. The candidate learning rates of AdaGrad are selected from $\{0.1, 0.05, 0.01, 5e-3, 1e-3\}$. For the initial accumulator value, we choose the default value 0. The best learning rate of AdaGrad is 0.01 for the three experiments.

Adam, AMSGrad and AdaBound. The candidate learning rates of Adam, AMSGrad and AdaBound are selected from $\{0.01, 5e-3, 1e-3, 5e-4, 1e-4\}$. We turn over β_1 values of $\{0.9, 0.99\}$ and β_2 values of $\{0.99, 0.999\}$. We use the default values of other hyperparameters. The best learning rate is $1e-3$ for all experiments.

SEHB. The candidate learning rates of SEHB are selected from $\{0.1, 0.05, 0.03, 0.01, 1e-3\}$. We use the default value of c and γ_t. The best learning rate is 0.1 for fully connected neural network on MNIST dataset, 0.01 for ResNet-50 on CIFAR-10 dataset and 0.03 for DenseNet-121 on CIFAR-10 dataset.

4.2.2 FULLY CONNECTED NEURAL NETWORK ON MNSIT

Fully connected neural network we using in the experiment has only one hidden layer. We run the fully connected neural network with Adam, AdaGrad, AEGD, SHB and our method SEHB on MNIST dataset. The total number of epochs is 100 and the minibatch size is set to 128. Figure 2 shows the empirical results.

Note that all optimization algorithms achieve a training loss approaching 0 and have a test error below 2%. We find that our method SEHB and Adam achieve slightly better performance than other methods on the test set. However, SEHB have a smoother and faster convergence process than Adam and generalize more stably than Adam.

4.2.3 RESNET-50 AND DENSENET-121 ON CIFAR-10 DATASET

CIFAR-10 is a more complex dataset than MNIST. We use more advanced and powerful deep convolution neural networks, including ResNet-50 and DenseNet-121, to test various optimization methods in this classification task on CIFAR-10 dataset. We employ the fixed budget of 200 epochs, set the minibatch size to 128 and adjust the learning rates through dividing them by 10 after 150 epochs. Finally, we apply weight decay scheme into optimization methods. Figure 3 shows the empirical results.

As is expected, the overall performance of each algorithm on ResNet-50 is similar to that on DenseNet-121. We note that SEHB shows the best generalization performance on both ResNet-50 and DenseNet-121. For ResNet-50, the best error of SEHB is lower than that of SHB with a margin 1.8% and is lower than that of Adam with a margin 1.5%; For DenseNet-121, the improvement of SEHB compared to SHB and Adam is 0.9% and 2% respectively. We find that classical adaptive methods show rapid descent in the early period of training such as AdaBound and AdaGrad. However, they show mediocre generalization ability on the test set. Almost all methods benefit from the learning rate adjustment except AdaGrad and after the adjustment, their performance on the test set have a significant improvement. The empirical results show that despite the relative bad generalization ability of adaptive methods, our method SEHB seems to overcome the drawback and achieve even better generalization performance than SHB.

4.2.4 Analysis

We select the classical numerical experiment and popular tasks from computer vision to investigate the efficacy of our proposed method. Empirical results demonstrate that our method SEHB show the best generalization performance. As is shown above, SEHB exceeds all other methods greatly with respect to test error at the ending of training. It indicates that SEHB, as a new variant of adaptive methods, also has a excellent generalization performance which is comparable and even better than SHB. It seems that SEHB overcomes the drawback of adaptive methods which typically have worse generalization ability. Our aim to combine adaptively learning rate adjustment with SHB is proved to be feasible and successful.

5 Conclusion

By investigating AEGD and IEQ, we find that their update rule is equivalent to Euler's method. Inspired by this, we propose a new variant of SHB, called SEHB, by adding adaptive learning rate adjustment into SHB. SEHB shows rapid convergence and better generalization ability than SHB, which is far different from current adaptive methods like Adam and AMSGrad. We argue that SEHB provides a new way to adaptively adjust learning rates and realizes the combination of adaptive methods with SHB just like Adam. The empirical results show that SEHB is worth further research.

Despite excellent performance of SEHB, there still remains several problems to explore. First, SEHB employs Euler's method as the adaptive adjustment of learning rates which is different from the mainstream adaptive methods for deep learning. The reason for the effectiveness of this way of adjusting remains unclear. Furthermore, AMSGrad shows that very small values of adaptive learning rates may result in worse performance and even non-convergence, thus the effectiveness of adding the control of adaptive learning rates for SEHB needs more explaination. Finally, the reason for better generalization performance of SEHB than SHB is still uncertain and worth further exploration.

REFERENCES

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *Journal of Machine Learning Research (JMLR)*, 2011.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey E Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 6645–6649, 2013.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Nitish Shirish Keskar and Richard Socher. Improving generalization performance by switching from Adam to SGD. In *CoRR*, 2017.

Diederik P Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

Alex Krizhevsky and Geoffrey E. Hinton. Learning multiple layers of features from tiny images. In *Technical report*, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems (NIPS)*, pp. 1097–1105, 2012.

Yann Lecun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pp. 2278–2324, 1998.

Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2019.

Yurii Nesterov. A method of solving a convex programming problem with convergence rate O(1/sqr(k)). In *Soviet Mathematics Doklady*, pp. 27:372–376, 1983.

Boris. T. Polyak. Some methods of speeding up the convergence of iteration methods. In *USSR Computational Mathematics and Mathematical Physics*, pp. 4:791–803, 1964.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of Adam and beyond. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2018.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning(ICML)*, pp. 1139–1147, 2013.

Xuping Tan. AEGD: adaptive gradient decent with energy. 2020. URL `https://github.com/txping/AEGD`.

Tijmen Tieleman and Geoffrey Hinton. RMSprop: Divide the gradient by a running average of its recent magnitude. In *COURSERA: Neural networks for machine learning*, 2012.

Xiaofeng Yang. Linear, first and second order and unconditionally energy stable numerical schemes for the phase field model of homopolymer blends. *JCP*, pp. 302:509–523, 2016.

Matthew D. Zeiler. ADADELTA: An adaptive learning rate method. In *CoRR*, 2012.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, pp. 267–280, 2003.

APPENDIX

A CONVERGENCE PROOF

The proof of Theorem 1 is as follows:

Proof. Let $\theta^* = \arg\min_\theta f_t(\theta)$, we have

LEMMA 1. If a function $f : \mathbb{R}^d \to \mathbb{R}$ is convex, then for all $\theta_1, \theta_2 \in \mathbb{R}^d$,

$$f(\theta_2) \geq f(\theta_1) + \nabla f(\theta_1)^T(\theta_2 - \theta_1). \tag{8}$$

Actually, it is an equivalent definition of convex functions.

According to Lemma 1, we have

$$\begin{aligned} f_t(\theta_t) - f_t(\theta^*) &\leq \left\langle \nabla f_t(\theta_t), \theta_t - \theta^* \right\rangle \\ &= \sum_{i=1}^{d} \left\langle \nabla f_t(\theta_{t,i}), \theta_{t,i} - \theta^*_{,i} \right\rangle. \end{aligned} \tag{9}$$

Recall our update formula:

$$\begin{aligned} \theta_{t+1} - \theta_t &= -2\eta_t r_{t+1} v_{t+1} \\ &= -2\eta_t r_{t+1}(\gamma v_t + \nabla g(\theta_t)). \end{aligned} \tag{10}$$

For i-th coordinate of θ,

$$\theta_{t+1,i} - \theta_{t,i} = -2\eta r_{t+1,i}(\gamma v_{t,i} + \nabla g(\theta_{t,i})), \tag{11}$$

and then

$$\begin{aligned} (\theta_{t+1,i} - \theta^*_{,i})^2 &= (\theta_{t+1,i} - \theta_{t,i} + \theta_{t,i} - \theta^*_{,i})^2 \\ &= (\theta_{t,i} - \theta^*_{,i})^2 + 2(\theta_{t+1,i} - \theta_{t,i})(\theta_{t,i} - \theta^*_{,i}) + (\theta_{t+1,i} - \theta_{t,i})^2 \\ &= (\theta_{t,i} - \theta^*_{,i})^2 - 4\eta r_{t+1,i}(\gamma_t v_{t,i} + \nabla g(\theta_{t,i}))(\theta_{t,i} - \theta^*_{,i}) + (-2\eta_t r_{t+1,i} v_{t+1,i})^2. \end{aligned} \tag{12}$$

Thus,

$$\begin{aligned} \nabla f(\theta_{t,i})(\theta_{t,i} - \theta^*_{,i}) =& \frac{2g(\theta_{t,i})}{4\eta_t r_{t+1,i}}((\theta_{t,i} - \theta^*_{,i})^2 - (\theta_{t+1,i} - \theta^*_{,i})^2) - 2g(\theta_{t,i})\gamma_t v_{t,i}(\theta_{t,i} - \theta^*_{,i}) \\ &+ 2g(\theta_{t,i})\eta_t r^2_{t+1,i} v^2_{t+1,i}. \end{aligned} \tag{13}$$

Because $a^2/2 + b^2/2 \geq ab$, we have

$$\begin{aligned} \nabla f(\theta_{t,i})(\theta_{t,i} - \theta^*_{,i}) \leq& 2g(\theta_{t,i})(\frac{1}{4\eta r_{t+1,i}}((\theta_{t,i} - \theta^*_{,i})^2 - (\theta_{t+1,i} - \theta^*_{,i})^2) \\ &+ \gamma(v^2_{t,i}/2 + (\theta_{t,i} - \theta^*_{,i})^2/2) + \eta r^2_{t+1,i} v^2_{t+1,i}). \end{aligned} \tag{14}$$

Before proving the regret bound, we prove the following Lemma 2, Lemma 3 and Lemma 4:

LEMMA 2. $r_{t,i}$ is strictly decreasing and $r_{t,i} > 0$.

proof. Because $r_{1,i} = \sqrt{f(\theta_1) + c} > 0$, we have

$$r_{t,i} = \frac{r_{t-1,i}}{1 + 2\eta_t G^2_{t,i}} = \frac{r_{1,i}}{\prod_{j=1}^{t}(1 + 2\eta_j G^2_{j,i})}, \tag{15}$$

where $G_t = \nabla g_t(\theta_t)$ and $G_{t,i}$ is the i-th coordinate of G_t. Obviously, $1 + 2\eta_j G^2_{j,i} > 1$, for $j \in \{1, 2, \cdots, t\}$. This proves the Lemma 2.

LEMMA 3.

$$|v_{t+1,i}| \le \frac{G_\infty \gamma_t}{1-\gamma_t}. \tag{16}$$

proof.

$$\begin{aligned}
|v_{t+1,i}| &= \gamma_t v_{t,i} + \nabla g_t(\theta_{t,i}) \\
&\le \sum_{j=1}^{t} \gamma_t^{t+1-j} \nabla g_j(\theta_{j,i}) \\
&\le G_\infty \sum_{j=1}^{t} \gamma_t^{t+1-j} \\
&\le \frac{G_\infty \gamma_t}{1-\gamma_t}.
\end{aligned} \tag{17}$$

LEMMA 4

$$\begin{aligned}
\sum_{t=1}^{T}\sum_{i=1}^{d} \gamma_t v_{t,i}^2 &\le \frac{dG_\infty^2 \gamma^3}{\lambda^2(1-\gamma)^2(1-\lambda^2)}, \\
\sum_{t=1}^{T}\sum_{i=1}^{d} \eta_t v_{t+1,i}^2 &\le \frac{d\eta G_\infty^2 \gamma^2}{(1-\gamma)^2(1-\lambda^2)}.
\end{aligned} \tag{18}$$

proof.

$$\begin{aligned}
\sum_{t=1}^{T}\sum_{i=1}^{d} \gamma_t v_{t,i}^2 &\le \gamma d G_\infty^2 \sum_{t=1}^{T} \frac{\gamma_{t-1}^2}{(1-\gamma_{t-1})^2} \le dG_\infty^2 \gamma^3 \lambda^{-2} \sum_{t=1}^{T} \frac{\lambda^{2t-2}}{(1-\gamma)^2} \\
&= dG_\infty^2 \gamma^3 \lambda^{-2} \frac{\frac{1-\lambda^{2t}}{1-\lambda^2}}{(1-\gamma)^2} \le \frac{dG_\infty^2 \gamma^3}{\lambda^2(1-\gamma)^2(1-\lambda^2)}, \\
\sum_{t=1}^{T}\sum_{i=1}^{d} \eta_t v_{t+1,i}^2 &\le \eta d G_\infty^2 \sum_{t=1}^{T} \frac{\gamma_t^2}{(1-\gamma_t)^2} \le \eta d G_\infty^2 \gamma^2 \sum_{t=1}^{T} \frac{\lambda^{2t-2}}{(1-\gamma)^2} \\
&= d\eta G_\infty^2 \gamma^2 \frac{\frac{1-\lambda^{2t}}{1-\lambda^2}}{(1-\gamma)^2} \le \frac{d\eta G_\infty^2 \gamma^2}{(1-\gamma)^2(1-\lambda^2)}.
\end{aligned} \tag{19}$$

Finally, we upper the regret bound:

$$\begin{aligned}
R(T) &= \sum_{t=1}^{T} (f_t(\theta_t) - f_t(\theta^*)) \\
&= \sum_{t=1}^{T}\sum_{i=1}^{d} \langle \nabla f_t(\theta_{t,i}), \theta_{t,i} - \theta_{,i}^* \rangle.
\end{aligned} \tag{20}$$

According the update rule and Equation 14, we have

$$\begin{aligned}
R(T) \leq& \sum_{t=1}^{T}\sum_{i=1}^{d}\frac{2g(\theta_{t,i})}{4\eta_t r_{t+1,i}}((\theta_{t,i}-\theta^*_{,i})^2-(\theta_{t+1,i}-\theta^*_{,i})^2)+\sum_{t=1}^{T}\sum_{i=1}^{d}2g(\theta_{t,i})(\gamma_t(v^2_{t,i}/2+(\theta_{t,i}-\theta^*_{,i})^2/2)\\
&+\eta_t r^2_{t+1,i}v^2_{t+1,i})\\
=& \sum_{t=1}^{T}\sum_{i=1}^{d}(\frac{2g(\theta_{t,i})}{4\eta_t r_{t+1,i}}+\gamma_t g(\theta_{t,i}))(\theta_{t,i}-\theta^*_{,i})^2-\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{2g(\theta_{t,i})}{4\eta_t r_{t+1,i}}(\theta_{t+1,i}-\theta^*_{,i})^2\\
&+\sum_{t=1}^{T}\sum_{i=1}^{d}2g(\theta_{t,i})(\gamma_t v^2_{t,i}/2+\eta_t r^2_{t+1,i}v^2_{t+1,i})\\
\leq& \sum_{i=1}^{d}\frac{2g(\theta_{1,i})}{4\eta_T r_{2,i}}(\theta_{1,i}-\theta^*_{,i})^2+\sum_{t=1}^{T}\sum_{i=1}^{d}\gamma_t g(\theta_{t,i})(\theta_{t,i}-\theta^*_{,i})^2\\
&+\sum_{t=2}^{T}\sum_{i=1}^{d}(\frac{2g(\theta_{t,i})}{4\eta_t r_{t+1,i}}-\frac{2g(\theta_{t-1,i})}{4\eta_t r_{t,i}})(\theta_{t,i}-\theta^*_{,i})^2+\sum_{t=1}^{T}\sum_{i=1}^{d}2g(\theta_{t,i})(\gamma_t v^2_{t,i}/2+\eta_t r^2_{t+1,i}v^2_{t+1,i}).
\end{aligned} \tag{21}$$

By applying the assumptions into the proof, we obtain

$$\begin{aligned}
R(T) \leq& D^2_\infty\sum_{i=1}^{d}(\frac{2g(\theta_{T,i})}{4\eta_T r_{T+1,i}}-\frac{2g(\theta_{1,i})}{4\eta_2 r_{2,i}})\\
&+D^2_\infty\sum_{i=1}^{d}\frac{2g(\theta_{1,i})}{4\eta_T r_{2,i}}+D^2_\infty\sum_{t=1}^{T}\sum_{i=1}^{d}\gamma_t g(\theta_{t,i})+\sum_{t=1}^{T}\sum_{i=1}^{d}2g(\theta_{t,i})(\gamma_t v^2_{t,i}/2+\eta_t r^2_{t+1,i}v^2_{t+1,i})\\
\leq& D^2_\infty\sum_{i=1}^{d}\frac{2g(\theta_{1,i})}{4\eta_T r_{2,i}}+D^2_\infty\sum_{t=1}^{T}\sum_{i=1}^{d}\gamma_t g(\theta_{t,i})+\sum_{t=1}^{T}\sum_{i=1}^{d}2g(\theta_{t,i})(\gamma_t v^2_{t,i}/2+\eta_t r^2_{t+1,i}v^2_{t+1,i})\\
&+g_\infty D^2_\infty\sum_{i=1}^{d}\frac{\sqrt{T}}{2\eta r_{T+1,i}}\\
\leq& D^2_\infty\sqrt{T}\sum_{i=1}^{d}\frac{g(\theta_{1,i})}{2\eta r_{2,i}}+g_\infty D^2_\infty\sum_{t=1}^{T}\sum_{i=1}^{d}\gamma_t+2g_\infty\sum_{t=1}^{T}\sum_{i=1}^{d}(\gamma_t v^2_{t,i}/2+\eta_t r^2_{t+1,i}v^2_{t+1,i})\\
&+g_\infty D^2_\infty\sum_{i=1}^{d}\frac{\sqrt{T}}{2\eta r_{T+1,i}}\\
\leq& D^2_\infty\sqrt{T}\sum_{i=1}^{d}\frac{g(\theta_{1,i})}{2\eta r_{2,i}}+g_\infty D^2_\infty\frac{d\gamma}{1-\lambda}+2g_\infty\sum_{t=1}^{T}\sum_{i=1}^{d}(\gamma_t v^2_{t,i}/2+\eta_t r^2_{t+1,i}v^2_{t+1,i})\\
&+g_\infty D^2_\infty\sum_{i=1}^{d}\frac{\sqrt{T}}{2\eta r_{T+1,i}}.
\end{aligned} \tag{22}$$

We apply Lemma 4 to the proof,

$$\begin{aligned}
R(T) &\leq g_\infty D_\infty^2 \sum_{i=1}^{d} \frac{\sqrt{T}}{2\eta r_{T+1,i}} + D_\infty^2 \sqrt{T} \sum_{i=1}^{d} \frac{g(\theta_{1,i})}{2\eta r_{2,i}} + \frac{g_\infty D_\infty^2 d\gamma}{1-\lambda} + \frac{g_\infty d G_\infty^2 \gamma^3}{\lambda^2(1-\gamma)^2(1-\lambda^2)} + \frac{2g_\infty^3 d G_\infty^2 \gamma^2 \eta}{(1-\gamma)^2(1-\lambda^2)} \\
&= g_\infty D_\infty^2 \sum_{i=1}^{d} \frac{\sqrt{T}}{2\eta r_{T+1,i}} + D_\infty^2 \sqrt{T} \sum_{i=1}^{d} \frac{g(\theta_{1,i})}{2\eta r_{2,i}} + \frac{g_\infty D_\infty^2 d\gamma}{1-\lambda} + \frac{g_\infty d G_\infty^2 \gamma^2 (\gamma + 2g_\infty^2 \lambda^2 \eta)}{\lambda^2(1-\gamma)^2(1-\lambda^2)} \\
&\leq D_\infty^2 \sqrt{T} g_\infty \sum_{i=1}^{d} (\frac{1}{2\eta r_{T+1,i}} + \frac{1}{2\eta r_{2,i}}) + \frac{g_\infty D_\infty^2 d\gamma}{1-\lambda} + \frac{g_\infty d G_\infty^2 \gamma^2 (\gamma + 2g_\infty^2 \lambda^2 \eta)}{\lambda^2(1-\gamma)^2(1-\lambda^2)} \\
&\leq \frac{1}{2\eta} D_\infty^2 \sqrt{T} g_\infty \sum_{i=1}^{d} (\frac{1}{r_{T+1,i}} + \frac{1}{r_{2,i}}) + \frac{g_\infty D_\infty^2 d\gamma}{1-\lambda} + \frac{g_\infty d G_\infty^2 \gamma^2 (\gamma + 2g_\infty^2 \lambda^2 \eta)}{\lambda^2(1-\gamma)^2(1-\lambda^2)}.
\end{aligned} \tag{23}$$

This is the regret bound of SEHB.

□