MOMENTUM IN MOMENTUM FOR ADAPTIVE OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Adaptive gradient methods, e.g., ADAM, have achieved tremendous success in machine learning. Employing adaptive learning rates according to the gradients, such methods are able to attain rapid training of modern deep neural networks. Nevertheless, they are observed to suffer from compromised generalization capacity compared with stochastic gradient descent (SGD) and tend to be trapped in local minima at an early stage during the training process. Intriguingly, we discover that the issue can be resolved by substituting the gradient in the second raw moment estimate term with its momentumized version in ADAM. The intuition is that the gradient with momentum contains more accurate directional information, and therefore its second moment estimation is a more preferable option for learning rate scaling than that of the raw gradient. Thereby we propose ADAM^3 as a new optimizer reaching the goal of training quickly while generalizing much better. We further develop a theory to back up the improvement in generalization and provide novel convergence guarantees for our designed optimizer. Extensive experiments on a variety of tasks and models demonstrate that ADAM^3 exhibits state-of-the-art performance and superior training stability consistently. Considering the simplicity and effectiveness of ADAM^3, we believe it has the potential to become a new standard method in deep learning. Code will be publicly available.

1 INTRODUCTION

Prevailing first-order optimization algorithms in modern machine learning can be classified into two categories. One is stochastic gradient descent (SGD) (Robbins & Monro, 1951), which is widely adopted due to its low memory cost and outstanding performance. SGDM (Sutskever et al., 2013) which incorporates the notion of momentum into SGD, has become the best choice for optimization in computer vision. The drawback of SGD(M) is that it scales the gradient uniformly in all directions, making the training slow, especially at the beginning, and fails to optimize complicated models well beyond Convolutional Neural Networks (CNN). The other type is adaptive gradient methods. Unlike SGD, adaptive gradient optimizers adapt the stepsize (a.k.a. learning rate) elementwise according to the gradient values. Specifically, they scale the gradient by the square roots of some form of the running average of the squared values of the past gradients. Popular examples include AdaGrad (Duchi et al., 2011), RMSprop (Tijmen Tieleman, 2012) and Adam (Kingma & Ba, 2015) etc. Adam, in particular, has become the default choice for many machine learning application areas. This is because compared to SGD(M), Adam is better at optimizing complex loss functions (Zhang et al., 2019; Kingma & Ba, 2015), e.g. those in deep learning.

Despite their fast speed in the early training phase, adaptive gradient methods, especially Adam, are found by studies (Wilson et al., 2017; Zhou et al., 2020) to be more likely to exhibit poorer generalization ability than SGD (M). This is discouraging because the ultimate goal of training in many machine learning tasks is to exhibit favorable performance during the testing phase. In recent years researchers have put much effort into mitigating the deficiencies of adaptive gradient algorithms. AMSGrad (Reddi et al., 2018b) is proposed to optimize loss functions empirically faster than Adam, and meanwhile, to fix the convergence problem in the original Adam (Kingma & Ba, 2015) paper, which also implies that AMSGrad theoretically converges faster than Adam. Yogi (Reddi et al., 2018a) takes the effect of batch size into consideration. M-SVAG (Balles & Hennig, 2018) transfers the variance adaptation mechanism from Adam to SGD. AdamW (Loshchilov & Hutter, 2017) changes the way in which L2 regularization is applied to Adam-alike algorithms for the first time.

SWATS (Keskar & Socher, 2017) switches from Adam to SGD throughout the training process via a hard schedule, and AdaBound (Luo et al., 2019) switches with a smooth transition by imposing dynamic bounds on stepsizes. More recently, RAdam (Liu et al., 2019) rectifies the variance of the adaptive learning rate by investigating the theory behind warmup heuristic (Popel & Bojar, 2018; Vaswani et al., 2017). AdaBelief (Zhuang et al., 2020) is a pioneering work that adapts stepsizes by the belief in the observed gradients and surpasses all the existing adaptive gradient methods in generalization performance. Nevertheless, most of these proposed variants, despite surpassing Adam in some scenarios, still generalize worse than SGD(M) on CNN-based vision tasks and their improvments over Adam are not significant enough yet. Till today, SGD and Adam are still the default options in machine learning, especially deep learning (Schmidt et al., 2021). Conventional rules for choosing optimizers are: Choose SGDM for Fully Connected Networks and CNNs, and Adam for Recurrent Neural Networks (RNN) (Cho et al., 2014; Hochreiter & Schmidhuber, 1997b), Transformers (Vaswani et al., 2017) and Generative Adversarial Networks (GAN) (Goodfellow et al., 2014). Based on the above observations, a natural question is:

Is there an adaptive gradient algorithm that can converge fast and meanwhile generalize well?

In this work, we are delighted to discover that simply replacing the gradient term in the second moment estimation term of Adam with its **momentumized** version can achieve this goal. Our idea comes from the origin of the Adam optimizer, which is a combination of RMSprop and SGDM. RMSprop scales the current gradient by the square root of the exponential moving average (EMA) of the squared past gradients, and Adam replaces the raw gradient in the numerator of the update term of RMSprop with its EMA form, i.e., with momentum. Since the EMA of the gradient is a more accurate estimation of the appropriate direction to descent, we consider putting it in the second moment estimation term as well. We find such operation makes the optimizer more suitable for the typical loss curvature and can theoretically converge to minima that generalize better. Extensive experiments on a broad range of tasks and models indicate that: without bells and whistles, our proposed optimizer can be as good as SGDM on vision problems and outperforms all the SOTA optimizers in other tasks meanwhile maintaining fast convergence speed. Our algorithm is efficient with no additional memory cost and applicable to a wide range of scenarios in machine learning and deep learning. More importantly, AdaM^3 requires little effort in hyperparameter tuning, and the default parameter setting for adaptive gradient methods works well consistently in our algorithm.

Notation We use t, T to symbolize the current and total iteration number in the optimization process. $\theta \in \mathbb{R}^d$ denotes the model parameter and $f(\theta) \in \mathbb{R}$ denotes the loss function. We further use θ_t to denote the parameter at step t and f_t to denote the noisy realization of f at time t because of the mini-batch stochastic gradient mechanism. g_t denotes the t-th time gradient and α denotes stepsize. m_t, v_t represent the EMA of the gradient and the second moment estimation term at time t of adaptive gradient methods respectively. ϵ is a small constant number added in adaptive gradient methods to refrain the denominator from being too close to zero. β_1, β_2 are the decaying parameter in the EMA formulation of m_t and v_t correspondingly. For any vectors $a, b \in \mathbb{R}^d$, we employ $\sqrt{a}, a^2, |a|, a/b, a \geq b, a \leq b$ for elementwise square root, square, absolute value, division, greater or equal to, less than or equal to respectively. For any $1 \leq i \leq d$, $\theta_{t,i}$ denotes the i-th element of θ_t. Given a vector $x \in \mathbb{R}^d$, we use $\|x\|_2$ to denote its l_2-norm and $\|x\|_\infty$ to denote its l_∞-norm. The symbol $\lesssim$ means the order of the LHS is less than the order of the RHS, i.e., LHS/RHS $\to 0$ when $T \to \infty$. $\beta_{1,t}$ denote the value of β_1 at step t and α_t denotes the value of α at step t.

2 ALGORITHM

Preliminaries & Motivation Omitting the debiasing operation and the damping term ϵ, the adaptive gradient methods can be generally written in the following form:

$$\theta_{t+1} = \theta_t - \alpha \frac{m_t}{\sqrt{v_t}}. \tag{1}$$

Here m_t, v_t are called the first and second moment estimation terms. When $m_t = g_t$ and $v_t = 1$, equation 1 degenerates to the vanilla SGD. Rprop (Riedmiller & Braun, 1993) is the pioneering work using the notion of adaptive learning rate, in which $m_t = g_t$ and $v_t = g_t^2$. Actually, it is equivalent to only using the sign of gradients for different weight parameters. RMSprop (Tijmen Tieleman, 2012) forces the number divided to be similar for adjacent mini-batches by incorporating momentum

Algorithm 1 AdaM^3 (ours). All mathematical operations are element-wise.

1: **Initialization** : Parameter initialization θ_0, step size α, damping term ϵ, $m_0 \leftarrow 0, v_0 \leftarrow 0, t \leftarrow 0$
2: **while** θ_t not converged **do**
3: $t \leftarrow t + 1$ ▷ Updating time step
4: $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ ▷ Acquiring gradient at time t
5: $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$ ▷ EMA of gradients
6: $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) m_t{}^2 + \epsilon$ ▷ EMA of EMA of gradients plus damping term
7: $\widehat{m_t} \leftarrow m_t / (1 - \beta_1^t)$ ▷ Bias correction of first moment estimation
8: $\widehat{v_t} \leftarrow v_t / (1 - \beta_2^t)$ ▷ Bias correction of second moment estimation
9: $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / \sqrt{\widehat{v_t}}$ ▷ Updating parameters
10: **end while**

acceleration into v_t. Adam (Kingma & Ba, 2015) is built upon RMSprop in which it turns g_t into momentumized version. Both RMSprop and Adam boost their performance thanks to the smoothing property of EMA. Due to the fact that the EMA of the gradient is a more accurate estimation than the raw gradient, we deem that there is no reason to use g_t in lieu of m_t in the second moment estimation term v_t. Therefore we propose to replace the g_is in v_t of Adam with its EMA version m_is, which further smooths the EMA. Hence our v_t turns into the EMA of the square of the EMA of the past gradients. The comparison of the classic optimizers and ours is summarized in Tab. 1.

Table 1: Comparison of AdaM^3 and classic adaptive gradient methods in m_t and v_t in equation 1.

Optimizer	m_t	v_t
SGD	g_t	1
Rprop	g_t	g_t^2
RMSprop	g_t	$(1-\beta_2)\sum_{i=1}^{t}\beta_2^{t-i}g_i^2$
Adam	$(1-\beta_1)\sum_{i=1}^{t}\beta_1^{t-i}g_i$	$(1-\beta_2)\sum_{i=1}^{t}\beta_2^{t-i}g_i^2$
Ours	$(1-\beta_1)\sum_{i=1}^{t}\beta_1^{t-i}g_i$	$(1-\beta_2)\sum_{i=1}^{t}\beta_2^{t-i}\boldsymbol{m}_i{}^2$

Detailed Algorithm The detailed procedure of our proposed optimizer is displayed in Algorithm 1. There are two major modifications based on Adam, which are marked in red and blue, respectively. One is that we replace the g_t in v_t of Adam with m_t, which is the momentumized gradient. Hence we name our proposed optimizer as *AdaM*^{3}, where the *M*^{3} can be interpreted as either the three-fold EMA mechanism (two-fold EMA in the denominator v_t and one EMA in the numerator m_t) or the **M**o**M**entu**M**ized gradient in v_t. The other is the location of ϵ (in Adam ϵ is added after $\sqrt{\cdot}$ in line 9 of Alg.1). We discover that moving the location of ϵ term from outside the radical symbol to inside can consistently enhance performance. To the best of our knowledge, our method is the first attempt to put momentumized gradient in the second moment estimation term of adaptive gradient methods.

3 WHY ADAM^3 OVER ADAM?

3.1 ADAM^3 IS MORE SUITABLE FOR TYPICAL LOSS CURVATURE

In this section, we show that AdaM^3 can converge to (global) minima faster than Adam does via illustration. The left part of Figure 1 is the process of optimization from a plateau to a basin area, where a global optimum is assumed to exist. The right part is the zoomed-in version of the situation near the minimum, where we have some peaks and valleys. This phenomenon commonly takes place in optimization since there is only one global minimum with many local minima surrounding (Hochreiter & Schmidhuber, 1997a; Keskar et al., 2017).

Benefits of Substituting g_t with m_t. We first explain how substituting m_t for g_t in the preconditioner v_t can improve training via decomposing the trajectory of parameter point along the loss curve. **1)** In area A, the parameter point starts to slide down the curve, and $|g_t|$ begins to enlarge abruptly. So the actual stepsize $\alpha/\sqrt{v_t}$ is small for Adam. However, the absolute value of the momentumized gradient m_t is small since it is the EMA of the past gradients, making $\alpha/\sqrt{v_t}$ still large for AdaM^3. Hence AdaM^3 can maintain a higher training speed than Adam in this changing corner of the loss curve, which is what an optimal optimizer should do. **2)** In area B, since the exponential moving average decays the impact of past gradients exponentially w.r.t. t, the magnitude of the elements of m_t will gradually become as large as g_t. **3)** In area C, when the parameter approaches the basin, the magnitude of g_t decreases, making the stepsizes of Adam increase immediately. In

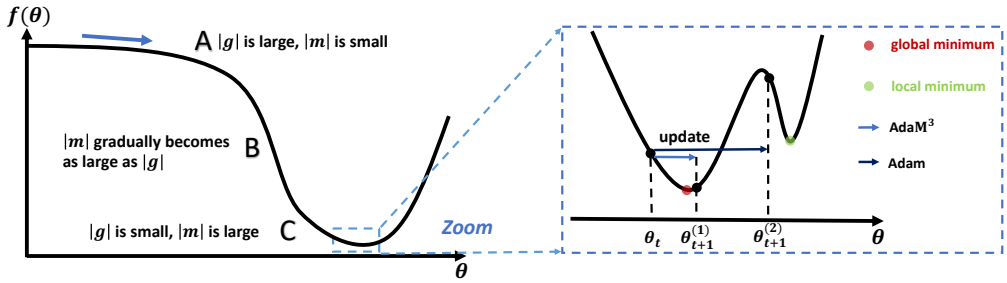


Figure 1: Illustration of the optimization process of Adam and AdaM^3. A typical loss curve can be composed to three areas: **A)** transition from a plateau to a downgrade; **B)** a steep downgrade; **C)** from downgrade to entering the basin containing the optimum. An ideal optimizer is expected to sustain large stepsize before reaching the optimum and reduce its stepsize near the optimum. Compared to Adam, AdaM^3 can adapt the effective stepsize more appropriately along the loss curve and maintain a smaller stepsize near the convergence, contributing to stable training and better convergence.

contrast, the stepsize of AdaM^3 is still comparatively small as $|m_t|$ is still much larger than $|g_t|$, which is desired for an ideal optimizer. Small stepsize near optimum is beneficial to convergence and stability (Luo et al., 2019; Zhuang et al., 2020) and a concrete illustration is given in the right part of Figure 1. If the stepsize is too large (e.g. in Adam), the weight parameter θ_t may rush to $\theta_{t+1}^{(2)}$ and miss the global optimum ($\theta_{t+1}^{(1)}$). Contrarily, small stepsize can guarantee the parameter to be close to the global minimum even if there may be tiny oscillations within the basin before the final convergence. The right part of Figure 1 mainly shows that large stepsize near convergence inevitably leads to undesirable training instability.

Benefits of Changing the Location of ϵ. Next, we elaborate on why putting ϵ under the $\sqrt{\cdot}$ is beneficial. We denote the debiased second moment estimation in AdaM^3 as $\widehat{v}_t$ and the second moment estimation term without ϵ as $\widehat{v}_t'$. By simple calculation, we have

$$\widehat{v}_t = \big((1-\beta_2)/(1-\beta_2^t)\big) \cdot \sum_{i=1}^{t} \beta_2^{t-i} m_i^2 + \frac{\epsilon}{1-\beta_2},$$

$$\widehat{v}_t' = \big((1-\beta_2)/(1-\beta_2^t)\big) \cdot \sum_{i=1}^{t} \beta_2^{t-i} m_i^2.$$

Hence we have $\widehat{v}_t = \widehat{v}_t' + \epsilon/(1-\beta_2)$. Then the actual stepsizes are $\alpha/(\sqrt{\widehat{v}_t' + \epsilon/(1-\beta_2)})$ and $\alpha/(\sqrt{\widehat{v}_t'}+\epsilon)$ for ϵ under $\sqrt{\cdot}$ and ϵ out of $\sqrt{\cdot}$ in AdaM^3 respectively. In the final stage of optimization, $\widehat{v}_t'$ is very close to 0 (because the values of gradients are near 0) and far less than ϵ hence the actual stepsizes can be approximately written as $\sqrt{1-\beta_2}\alpha/\sqrt{\epsilon}$ and α/ϵ. As ϵ usually takes very tiny values ranging from 10^{-8} to 10^{-16} and β_2 usually take values that are extremely close to 1 (usually 0.999), we have $\sqrt{1-\beta_2}\alpha/\sqrt{\epsilon} \ll \alpha/\epsilon$ [1]. Therefore we may reasonably come to the conclusion that after moving ϵ term into the radical symbol, AdaM^3 further reduces the stepsizes when the training is near minima, which contributes to enhancing convergence and stability as we have discussed above.

3.2 AdaM^3 Converges to Minima that Generalize Better

The outline of Adam and our proposed AdaM^3 can be written in the following unified form:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \quad v_t = \beta_2 v_{t-1} + (1-\beta_2) k_t^2,$$
$$\theta_{t+1} = \theta_t - \alpha\, m_t \Big/ \left((1-\beta_1^t) \sqrt{v_t/(1-\beta_2^t)} \right). \tag{2}$$

where k_t equalts g_t in Adam and m_t in AdaM^3. We introduce some definitions before proceeding.

Definition 1 (Symmetric α-stable distribution (Lévy & Lévy, 1954)). Let X_1 and X_2 be independent copies of a random variable X. Then X is said to be stable if for any constants $a > 0$ and $b > 0$ the random variable $aX_1 + bX_2$ has the same distribution as $cX + d$ for some constants $c > 0$ and d. Denote $\psi(t)$ as the characteristic function of the distribution of X, then it can be written as

[1] Note that even for different ϵ values we still have $\sqrt{1-\beta_2}\alpha/\sqrt{\epsilon_1} < \alpha/\epsilon_2$ as $\epsilon_1, \epsilon_2 \in [10^{-16}, 10^{-8}]$.

$\psi(t;\alpha,\beta,c,\mu) = \exp(it\mu - |ct|^{\alpha}(1 - i\beta\text{sgn}(t)\Phi))$, where $\text{sgn}(t)$ denotes the sign of t and

$$\Phi = \begin{cases} \tan\frac{\pi\alpha}{2} & \alpha \neq 1 \\ -\frac{2}{\pi}\log|t| & \alpha = 1. \end{cases}$$

$\mu \in \mathbb{R}$ is a shift parameter and $\beta \in [-1, 1]$. When $\beta = 0$ the distribution is symmetric about μ and is called (Levy) symmetric alpha-stable distribution, often abbreviated as $S\alpha S$ distribution.

Inspired by a line of work Pavlyukevich (2011); Simsekli et al. (2019); Zhou et al. (2020), we can consider equation 2 as a discretization of a continuous-time process and reformulate it as its corresponding Lévy-driven stochastic differential equation (SDE). Assuming that the gradient noise $\zeta_t = g_t - \nabla f(\theta_t)$ is centered symmetric $\widetilde{\alpha}$-stable ($\mathcal{S}\widetilde{\alpha}\mathcal{S}$) distributed with covariance matrix Σ_t possessing a heavy-tailed signature ($\widetilde{\alpha} \in (0, 2]$), then we can derive the Lévy-driven SDE of equation 2:

$$d\theta_t = -q_t R_t^{-1} m_t dt + \upsilon R_t^{-1}\Sigma_t dL_t, \tag{3}$$

$$dm_t = (1-\beta_1)(\nabla f(\theta_t) - m_t),\ dv_t = (1-\beta_2)(k_t^2 - v_t), \tag{4}$$

where $R_t = \text{diag}(\sqrt{v_t/(1-\beta_2^t)}), \upsilon = \alpha^{1-1/\widetilde{\alpha}}, q_t = 1/(1-\beta_1^t)$ and L_t is the $\widetilde{\alpha}$-stable Lévy motion with independent components. We are interested in the local stability of the optimizers and therefore we suppose process equation 4 is initialized in a local basin $\mathbf{\Omega}$ with a minimum θ^* (w.l.o.g., we assume $\theta^* = \mathbf{0}$). To investigate the escaping behavior of θ_t, we first introduce two technical definitions.

Definition 2 (**Radon Measure** (Simon et al., 1983))**.** If a measure $m(\cdot)$ defined on the σ-algebra of Borel sets of a Hausdorff topological space X is 1) inner regular on open sets, 2) outer regular on all Borel sets, and 3) finite on all compact sets, then the measure is called a *Radon measure*.

Definition 3 (**Escaping Time & Escaping Set).** We define *Escaping Time* $\Gamma := \inf\{t \geq 0 : \theta_t \notin \mathbf{\Omega}^{-\upsilon^\gamma}\}$, where $\mathbf{\Omega}^{-\upsilon^\gamma} = \{y \in \mathbf{\Omega} : \text{dis}(\partial\mathbf{\Omega}, y) \geq \upsilon^\gamma\}$ and $\gamma > 0$ is a constant. We define *Escaping Set* $\Upsilon := \{y \in \mathbb{R}^d : R_{\theta^*}^{-1}\Sigma_{\theta^*} y \notin \mathbf{\Omega}^{-\upsilon^\gamma}\}$, where $\Sigma_{\theta^*} = \lim_{\theta_t\to\theta^*}\Sigma_t, R_{\theta^*} = \lim_{\theta_t\to\theta^*} R_t$.

We impose some standard assumptions before studying the relationship between Γ and Υ.

Assumption 1. f is non-negative with an upper bound, and locally μ-strongly convex in $\mathbf{\Omega}$, i.e., for any compact and convex $K \subset \Omega$, f is strongly convex on K.

Assumption 2. There exists some constant $L > 0$, s.t. $\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2, \forall x, y$.

Remark 1. Assumption 1 and 2 impose some standard assumptions of stochastic optimization (Ghadimi & Lan, 2013; Johnson & Zhang, 2013).

Assumption 3. We assume that $\int_0^\Gamma \langle \nabla f(\theta_t)/(1 + f(\theta_t)), q_t R_t^{-1} m_t\rangle\, dt \geq 0$ a.e., and $\beta_1 \leq \beta_2 \leq 2\beta_1$. We further suppose that there exist $\upsilon_-, \upsilon_+ > 0$ s.t. each coordinate of $\sqrt{v_t}$ can be uniformly bounded in (υ_-, υ_+) and there exist $\tau_m, \tau > 0$ s.t. $\|m_t - \widehat{m}_t\|_2 \leq \tau_m \left\|\int_0^{t-}(m_x - \widehat{m}_x)\, dx\right\|_2$ and $\|\widehat{m}_t\|_2 \geq \tau \left\|\nabla f(\widehat{\theta}_t)\right\|_2$, where $\widehat{m}_t$ and $\widehat{\theta}_t$ are calculated by solving equation equation 4 with $\upsilon = 0$.

Remark 2. Assumption 3 requires the true gradient $\nabla f(\theta_t)$ to have similar directions to m_t as we assume the integral of their inner product to be non-negative along the iteration trajectory. This can be easily satisfied because m_t can be viewed as an estimate of $\nabla f(\theta_t)$. The distance assumption between m_t and $\widehat{m}_t$ can be easily fulfilled by their definitions.

Based on the above assumptions, we can prove that for algorithm of form equation 2, the expected escaping time is inversely proportional to the Radon measure of the escaping set:

Lemma 1. Under Assumptions 1-3, let $\upsilon^{\widetilde{\alpha}+1} = \Theta(\widetilde{\alpha})$ and $\ln(2\Delta/(\mu\upsilon^{1/3})) \leq 2\mu\tau(\beta_1 - \beta_2/4)/(\beta_1\upsilon_+ + \mu\tau)$, where $\Delta = f(\theta_0) - f(\theta^*)$. Then given any $\theta_0 \in \mathbf{\Omega}^{-2\upsilon^\gamma}$, for equation 4 we have

$$\mathbb{E}(\Gamma) = \Theta(\upsilon/m(\Upsilon)),$$

where $m(\cdot)$ is a non-zero Radon measure satisfying that $m(\mathcal{U}) < m(\mathcal{V})$ if $\mathcal{U} \subset \mathcal{V}$.

Remark 3. As larger set has larger volume, i.e., $V(\mathcal{U}) \leq V(\mathcal{V})$ if $\mathcal{U} \subset \mathcal{V}$, from Lemma 1 we have the escaping time is negatively correlated with the volume of the set Υ. Therefore, we can come to the conclusion that for both Adam and AdaM^3, if the basin $\mathbf{\Omega}$ is sharp which is ubiquitous during the early stage of training, Υ has a large Radon measure, which leads to smaller escaping time Γ. This means both Adam and AdaM^3 prefer relatively flat or asymmetric basin (He et al., 2019).

On the other hand, upon encountering a comparatively flat basin or asymmetric valley $\mathbf{\Omega}$, we are able to prove that AdaM^3 will stay longer inside. Before proceeding, we impose two mild assumptions.

Assumption 4. The l_∞ norm of ∇f is bounded by some constant G, *i.e.*, $\|\nabla f(x)\|_\infty \leq G, \forall x$.

Assumption 5. For AdaM^3, there exists $T_0 \in \mathbb{N}$ s.t., $\mathbb{E}(\zeta_t^2) \leq \beta_1 \mathbb{E}(m_{t-1}^2)/(2-\beta_1)$ when $t > T_0$.

Assumption 4 is a standard assumption in stochastic optimization (Guo et al., 2021; Reddi et al., 2018b; Savarese et al., 2021). As β_1 is always set as positive number close to 1, Assumption 5 basically requires that the gradient noise variance to be smaller than the second moment of m when t is very large. This is mild as 1) we can select mini-batch size to be large enough to satisfy it as the noise variance is inversely proportional to batch size (Bubeck, 2014). 2) The magnitudes of the variances of the stochastic gradients are usually much lower than that of the gradients (Faghri et al., 2020). In Fig. 2, we report the values of ζ_t^2 and $\beta_1 m_{t-1}^2/(2-\beta_1)$ of AdaM^3 on a 5-layer fully connected network with width 30. From Fig. 2, one can observe that ζ_t^2 is consistently lower than $\beta_1 m_{t-1}^2/(2-\beta_1)$ as iteration becomes larger, which further validates Assumption 5. Then we can come to the following result.

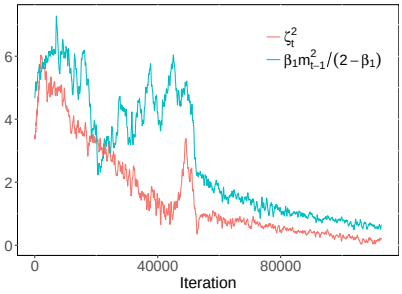


Figure 2: Empirical investigation of Assumption 5.

Proposition 1. Under Assumptions 1-5, upon encountering a comparatively flat basin or asymmetric valley $\mathbf{\Omega}$, we have

$$\mathbb{E}\left(\Gamma^{(\text{AdaM}^3)}\right) \geq \mathbb{E}\left(\Gamma^{(\text{Adam})}\right).$$

When falling into a flat/asymmetric basin, AdaM^3 is more stable than Adam and will not easily escape from it. Combining the aforementioned results and the fact that minima at the flat or asymmetric basins tend to exhibit better generalization performance (as observed in Keskar et al. (2017); He et al. (2019); Hochreiter & Schmidhuber (1997a); Izmailov et al. (2018); Li et al. (2018)), we are able to conclude that AdaM^3 is more likely to converge to minima that generalize better, which buttresses the empirical improvement of AdaM^3. The proofs in section 3.2 are given in Appendix A.

4 Convergence Analysis of AdaM^3

In this section, we establish the convergence theory for AdaM^3 under the non-convex object function condition. The convex convergence theory for AdaM^3 is provided in Appendix B.1 for completeness. We omit the two bias correction steps in the Algorithm 1 for simplicity, and the following analysis can be easily adapted to the de-biased version as well. When f is non-convex and lower-bounded, we derive the non-asymptotic convergence rate of AdaM^3 in the following theorem.

Theorem 1. Suppose that Assumptions 2 and 4 hold. We denote $b_{u,t} = \sqrt{(1-\beta_2)/(\epsilon - \epsilon\beta_2^t)} \leq b_{u,1}, b_{l,t} = 1/\left[\sqrt{G^2(1-\beta^T)^2 + \epsilon/(1-\beta_2)}(1-\beta_2^t)\right] \geq b_{l,T}$ where $\beta = \min_t \beta_{1,t}$. If there exists some $T_0 \lesssim 1/\alpha_T$, such that for all $t \geq T_0$ we have $\alpha_T \leq \alpha_t \leq (1-\beta_{1,t+1})\sqrt{b_{l,T}/(2L^2 b_{u,1}^3)}$ and $\alpha_t \leq b_{l,t}/(2Lb_{u,t}^2)$. With $\eta(T) \coloneqq \sum_{t=1}^T (1-\beta_{1,t})^2$, we have:

$$\mathbb{E}\left[\frac{1}{T+1}\sum_{t=0}^{T}\|\nabla f(\theta_t)\|_2^2\right] \leq \frac{1}{\alpha_T(T+1)}(Q_1 + Q_2\eta(T))$$

for some positive constants Q_1, Q_2 independent of d or T.

The conditions in Theorem 1 are reasonable, as in practice the momentum parameter for the first-order average $\beta_{1,t}$ is usually set as a large value, and meanwhile the step size α_t decays with time (Chen et al., 2019; Guo et al., 2021; Huang & Huang, 2021). Our non-convex analysis **improves** the current literature (Chen et al., 2019; Reddi et al., 2018b; Zhuang et al., 2020) in optimization for adaptive gradient method significantly in that we no longer require $\beta_{1,t}$ and $\alpha_t/\sqrt{v_t}$ to be monotonically decreasing (needed in Chen et al. (2019); Reddi et al. (2018b); Zhuang et al.

(2020)), which does not correspond to the realistic circumstances. In particular, we can use a setting with $(1-\beta_{1,t}) = 1/\sqrt{t}$ and $\alpha_t = \alpha/\sqrt{t}$ for some initial constant α to achieve the $O(\log(T)/\sqrt{T})$ convergence rate as in the following result.

Corollary 1. When $1-\beta_{1,t}$ and α_t are further chosen to be in the scale of $O(1/\sqrt{t})$ with all assumptions in Theorem 1 hold, AdaM^3 satisfies:

$$\mathbb{E}\left[\frac{1}{T+1}\sum_{t=0}^{T}\|\nabla f(\theta_t)\|_2^2\right] \leq \frac{1}{\sqrt{T}}(Q_1^* + Q_2^*\log(T)),$$

for some constants Q_1^*, Q_2^* similarly defined in Theorem 1.

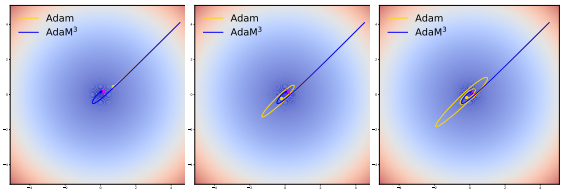
Corollary 1 manifests the $O(\log(T)/\sqrt{T})$ convergence rate of AdaM^3 under the non-convex object function case. We refer readers to the detailed proof in Appendix B.3.

5 Experiments

5.1 2D Toy Experiment on Sphere Function

We compare the optimization performance of AdaM^3 and Adam on 2D Sphere Function (bowl-shaped) (Dixon, 1978): $f(x) = x_1^2 + x_2^2$. We omit the damping term ϵ in both two algorithms, so the only difference is the m_t and g_t in the term v_t. We set the learning rate α of AdaM^3 as 0.1 and finetune the learning rate of Adam. We can observe from Fig. 3 that, on the one hand, when the α of Adam is the same as AdaM^3, Adam is much slower than our AdaM^3 in convergence; on the other hand, when we use a larger α on Adam ($\alpha = 0.5, 0.1$) it will oscillate much more violently. To summarize, the replacement of g_t with m_t in AdaM^3 makes the alteration of the learning rate smoother and more suitable (see analysis in Sec 3.1) for the sphere loss function. Despite the fact that this is only a toy experiment, such local behavior of AdaM^3 and Adam may shed light on their performance difference in complex deep learning tasks in the sequel of the paper, as any complicated real function can be approximated using the compositions of sphere functions (Yarotsky, 2017).

5.2 Deep Learning Experiments



(a) $\alpha_{\text{Adam}} = 0.1$. (b) $\alpha_{\text{Adam}} = 0.5$. (c) $\alpha_{\text{Adam}} = 1.0$.

Figure 3: Optimization trajectories of AdaM^3 and Adam on Sphere Function. The αs of AdaM^3 are 0.1.

We empirically investigate the performance of AdaM^3 in optimization, generalization and training stability. We conduct experiments on various modern network architectures for different tasks covering both vision and language processing area: **1)** image Classification on CIFAR-10 (Krizhevsky & Hinton, 2009) and ImageNet (Russakovsky et al., 2015) with CNN; **2)** language modeling on Penn Treebank (Marcus et al., 1993) dataset using Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997b); **3)** neural machine translation on IWSTL'14 DE-EN (Cettolo et al., 2014) dataset employing Transformer; **4)** Generative Adversarial Networks (GAN) on CIFAR-10. We compare AdaM^3 with seven state-of-the-art optimizers: SGDM (Sutskever et al., 2013), Adam (Kingma & Ba, 2015), AdamW (Loshchilov & Hutter, 2017), Yogi (Reddi et al., 2018a), AdaBound (Luo et al., 2019), RAdam (Liu et al., 2019) and AdaBelief (Zhuang et al., 2020). We perform a careful and extensive hyperparameter tuning (including learning rate, β_2, weight decay and ϵ) for all the optimizers compared in each experiment and report their best performance. The detailed tuning schedule is summarized in Appendix C. It is worth mentioning that in experiments we discover that setting $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999$ (the default setting for adaptive gradient methods in applied machine learing) works well in most cases. This elucidates that our optimizer is **tuning-friendly**, which reduces human labor and time cost and is crucial in practice. The mean results with standard deviations over 5 random seeds are reported in all the following experiments except ImageNet. Moreover, the pairwise hypothesis testing results between our method and other gradient methods using Paired Wilcoxon signed-rank test (exact form) (Wilcoxon, 1992) and Paired t-test (Shao, 2003) are provided in Appendix D.

Table 2: Test accuracy (%) of CNNs on CIFAR-10 dataset. The best in red and second best in blue.

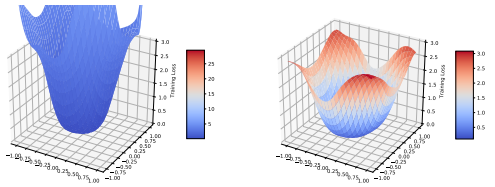
Architecture	Measure	Non-adaptive	Adaptive gradient methods						
		SGDM	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief	**AdaM^3**
VGGNet-16	Best	94.73$_{\pm 0.12}$	93.29$_{\pm 0.10}$	93.33$_{\pm 0.15}$	93.44$_{\pm 0.16}$	93.79$_{\pm 0.17}$	93.90$_{\pm 0.10}$	94.57$_{\pm 0.09}$	94.80$_{\pm 0.10}$
	Last	94.64$_{\pm 0.17}$	93.09$_{\pm 0.10}$	93.20$_{\pm 0.17}$	93.23$_{\pm 0.17}$	93.65$_{\pm 0.20}$	93.78$_{\pm 0.13}$	94.44$_{\pm 0.06}$	94.69$_{\pm 0.12}$
ResNet-34	Best	96.47$_{\pm 0.09}$	95.39$_{\pm 0.11}$	95.48$_{\pm 0.10}$	95.28$_{\pm 0.19}$	95.51$_{\pm 0.07}$	95.67$_{\pm 0.16}$	96.04$_{\pm 0.07}$	96.33$_{\pm 0.07}$
	Last	96.31$_{\pm 0.11}$	95.25$_{\pm 0.09}$	95.36$_{\pm 0.08}$	95.11$_{\pm 0.15}$	95.41$_{\pm 0.08}$	95.61$_{\pm 0.13}$	95.94$_{\pm 0.12}$	96.18$_{\pm 0.11}$
DenseNet-121	Best	96.19$_{\pm 0.17}$	95.35$_{\pm 0.09}$	95.52$_{\pm 0.14}$	94.98$_{\pm 0.13}$	95.43$_{\pm 0.12}$	95.82$_{\pm 0.19}$	96.09$_{\pm 0.14}$	96.30$_{\pm 0.12}$
	Last	96.04$_{\pm 0.16}$	95.25$_{\pm 0.11}$	95.37$_{\pm 0.14}$	94.89$_{\pm 0.13}$	95.31$_{\pm 0.17}$	95.73$_{\pm 0.21}$	95.95$_{\pm 0.16}$	96.20$_{\pm 0.12}$

Table 4: Test perplexity (↓) results of LSTMs on Penn Treebank dataset.

Layer #	SGDM	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief	**AdaM^3**
1	85.31$_{\pm 0.09}$	84.55$_{\pm 0.10}$	88.18$_{\pm 0.14}$	86.87$_{\pm 0.14}$	85.10$_{\pm 0.22}$	88.60$_{\pm 0.22}$	84.30$_{\pm 0.23}$	**80.82$_{\pm 0.19}$**
2	67.25$_{\pm 0.20}$	67.11$_{\pm 0.20}$	73.61$_{\pm 0.15}$	71.54$_{\pm 0.14}$	67.69$_{\pm 0.24}$	73.80$_{\pm 0.25}$	66.66$_{\pm 0.11}$	**64.85$_{\pm 0.09}$**
3	63.52$_{\pm 0.16}$	64.10$_{\pm 0.25}$	69.91$_{\pm 0.20}$	67.58$_{\pm 0.08}$	63.52$_{\pm 0.11}$	70.10$_{\pm 0.16}$	61.33$_{\pm 0.19}$	**60.08$_{\pm 0.11}$**

5.2.1 CNN FOR IMAGE CLASSIFICATION

CIFAR-10 We experimented with three prevailing deep CNN architectures: VGG-16 (Simonyan & Zisserman, 2015), ResNet-34 (He et al., 2016) and DenseNet-121 (Huang et al., 2017). In each experiment, we train the model for 200 epochs with batch size 128 and decay the learning rate by 0.2 at the 60-th, 120-th, and 160-th epoch. We employ the label smoothing technique (Szegedy et al., 2016), and the smoothing factor is chosen as 0.1. Both best epoch and last epoch test accuracy results are summarized in Tab. 2, and Fig. 5 in Appendix C displays the training and testing results of all the compared optimizers. As indicated, during training AdaM^3 can be as fast as other adaptive gradient methods, being much faster than SGDM, especially before the third learning rate annealing. In the testing phase, AdaM^3 can exhibit accuracy as good as SGDM (better than SGDM on VGGNet and DenseNet and slightly worse than SGDM on ResNet) and far exceeds all other adaptive gradient methods, including the recently proposed adaptive gradient optimizers including Yogi, AdaBound, RAdam and current SOTA AdaBelief.



(a) Adam. (b) AdaM^3.

Figure 4: Comparison of the basins around the convergent points of ResNet-34 trained by Adam and AdaM^3 on CIFAR-10.

We further visualize the basins of the convergent minima of the models trained by Adam and AdaM^3 respectively in Fig. 4. We train ResNet-34 on CIFAR-10 using random seed 0 for 200 epochs and depict the 3D loss landscapes along with two random directions (Li et al., 2018). The landscape of Adam is cropped along the z-axis to 3 for comparison. Obviously seen from Fig. 4, the basin of AdaM^3 is much more flat than that of Adam, which verifies our theoretical argument in Sec 3.2 that AdaM^3 is more likely to converge to and stay in flat minima of the training loss.

ImageNet To corroborate the effectiveness of our algorithm on more comprehensive dataset, we perform experiments on ImageNet ILSVRC 2012 dataset (Russakovsky et al., 2015) utilizing ResNet-18 as backbone. We execute each optimizer three times independently for 100 epochs utilizing cosine learning rate annealing strategy. As shown in Tab. 3, AdaM^3 far exceeds Adam in Top-1 test accuracy with nearly 6% accuracy gain and even performs better than SGDM.

Table 3: Top-1 test accuracy (%) on ImageNet.

SGDM	Adam	**AdaM^3**
70.73±0.07	64.99±0.12	**70.77±0.09**

5.2.2 LSTM FOR LANGUAGE MODELING

We implement LSTMs with 1 to 3 layers on the Penn Treebank dataset, where adaptive gradient methods are the main-stream choices (much better than SGD). In each experiment, we train the model for 200 epochs with a batch size of 20 and decay the learning rate by 0.1 at 100-th and 145-th epoch. Test perplexity (the lower the better) is summarized in Tab. 4. Clearly observed from

Table 6: FID score (↓) of GANs on CIFAR-10 dataset. † is reported in Zhuang et al. (2020).

Type of GAN	SGDM	Adam(W)	Yogi	AdaBound	RAdam	AdaBelief	**AdaM^3**
DCGAN	$223.77_{\pm 147.90}$	$52.39_{\pm 3.62}$	$63.08_{\pm 5.02}$	$126.79_{\pm 40.64}$	$48.24_{\pm 1.38}$	$47.25_{\pm 0.79}$	$\mathbf{46.66}_{\pm 1.94}$
SNGAN	$49.70_{\pm 0.41}$†	$13.05_{\pm 0.19}$†	$14.25_{\pm 0.15}$†	$55.65_{\pm 2.15}$†	$12.70_{\pm 0.12}$†	$12.52_{\pm 0.16}$†	$\mathbf{12.06}_{\pm 0.21}$
BigGAN	$16.12_{\pm 0.33}$	$7.24_{\pm 0.08}$	$7.38_{\pm 0.04}$	$14.81_{\pm 0.31}$	$7.17_{\pm 0.06}$	$7.22_{\pm 0.09}$	$\mathbf{7.16}_{\pm 0.05}$

Tab. 4, AdaM^3 achieves the lowest perplexity in all the settings and consistently outperforms other competitors by a considerable margin, with up to 3.48% perplexity reduction on 1-layer LSTM, 1.81% on 2-layer LSTM and 1.25% on 3-layer LSTM. The training and testing perplexity curve is given in Fig. 6 and 7 in Appendix C. Particularly on 2-layer and 3-layer LSTM, AdaM^3 maintains both the fastest convergence and the best performance, which substantiates its superiority.

5.2.3 Transformer for Neural Machine Translation

Table 5: BLEU score (↑) on IWSTL'14 DE-EN dataset.

SGDM	Adam	AdamW	AdaBelief	**AdaM^3**
28.22±0.21	30.14±1.39	35.62±0.11	35.60±0.11	**35.66±0.10**

Transformers have been the dominating architecture in NLP, and adaptive gradient methods are usually adopted for training owing to their stronger ability to handle attention-models (Zhang et al., 2019). To test the performance of AdaM^3 on transformer, we experiment on IWSTL'14 German-to-English with the Transformer *small* model adapting the code from the fairseq package.[2] We set the length penalty as 1.0, the beam size as 5, the initial warmup stepsize as 10^{-7} and the warmup updates iteration number to be 8000. We train the models for 55 epochs, and the results are reported according to the average of the last 5 checkpoints. As shown in Tab. 11, our optimizer achieves the highest average BLEU score with the lowest variance and exceeds the popular optimizer in NLP AdamW .

5.2.4 Generative Adversarial Network

Training of GANs is extremely unstable and challenging. To further study the optimization ability and numerical stability of AdaM^3, we experiment with three types of GANs: Deep Convolutional GAN (DCGAN) (Radford et al., 2015), Spectral normalized GAN (SNGAN) (Miyato et al., 2018) and BigGAN (Brock et al., 2019). For the generator and the discriminator network, we adopt CNN for DCGAN and ResNets for SNGAN and BigGAN. The BigGAN training is assisted with consistency regularization (Zhang et al., 2020) for better performance. We train DCGAN for 200000 iterations and the other two for 100000 iterations on CIFAR-10 with the batch size 64. The learning rates for the generator and the discriminator network are both set as 0.0002. For AdaM^3 all the other hyperparameters are set as default values. Experiments are run 5 times independently, and we report the mean and standard deviation of Frechet Inception Distance (FID, the lower, the better) (Heusel et al., 2017) in Tab. 6. From Tab. 6 it is reasonable to draw the conclusion that AdaM^3 outperforms all the best-tuned baseline optimizers for all the GAN architectures by a considerable margin, which validates its outstanding optimization ability and numerical stability. Here Adam equals AdamW because the optimal weight decay parameter value is 0.

6 Conclusion

In this work, we rethink the formulation of Adam and innovatively propose AdaM^3 as a new optimizer for machine learning adopting a novel momentum in momentum approach. We illustrate that AdaM^3 is more fit to the typical loss curve than Adam and theoretically demonstrate why AdaM^3 outperforms Adam in generalization. We further validate the superiority of AdaM^3 through extensive and a broad range of experiments. Our algorithm is simple and effective with four key advantages: **1)** maintaining fast convergence rate; **2)** closing the generalization gap between adaptive gradient methods and SGD(M); **3)** applicable to various tasks and models; **4)** introducing no additional parameters and easy to tune. The Combination of AdaM^3 with other techniques such as Nesterov's accelerated gradient (Dozat, 2016) may be of independent interest in the future.

[2] https://github.com/pytorch/fairseq

ETHICS STATEMENT

Our work follows all ethical standards and laws. All the experiments were conducted on publically available datasets, with no new data concerning human or animal subjects generated.

REPRODUCIBILITY STATEMENT

We adhere to ICLR reproducibility standards and provide all necessary information to reproduce our experimental and theoretical results. We ensure the reproducibility of our work through several ways, namely

- All the technical details and proofs in Section 3.2 are included in Appendix A.
- All the proofs in Section 4 are provided in Appendix B.
- The detailed hyperparameter tuning rules and configurations of all the reported experiments in Section 5 are given in Appendix C.
- Our source code will be made publicly available after the acceptance of the paper.

REFERENCES

Lukas Balles and Philipp Hennig. Dissecting adam: The sign, magnitude and variance of stochastic gradients. In *ICML*, 2018.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.

Sébastien Bubeck. Convex optimization: Algorithms and complexity. *arXiv preprint arXiv:1405.4980*, 2014.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, 2014.

Xiangyi Chen, Sijia Liu, Ruoyu Sun, and Mingyi Hong. On the convergence of a class of adam-type algorithms for non-convex optimization. In *ICLR*, 2019.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, 2014.

Laurence Charles Ward Dixon. The global optimization problem. an introduction. *Toward global optimization*, 1978.

Timothy Dozat. Incorporating nesterov momentum into adam. *ICLR Workshop*, 2016.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 2011.

Fartash Faghri, David Duvenaud, David J Fleet, and Jimmy Ba. A study of gradient variance in deep learning. *arXiv preprint arXiv:2007.04532*, 2020.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 2013.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *NeurIPS*, 2014.

Zhishuai Guo, Yi Xu, Wotao Yin, Rong Jin, and Tianbao Yang. On stochastic moving-average estimators for non-convex optimization. *arXiv preprint arXiv:2104.14840*, 2021.

Haowei He, Gao Huang, and Yang Yuan. Asymmetric valleys: Beyond sharp and flat local minima. *NeurIPS*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 2017.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 1997a.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997b.

Feihu Huang and Heng Huang. Biadam: Fast adaptive bilevel optimization methods. *arXiv preprint arXiv:2106.11396*, 2021.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *UAI*, 2018.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *NeurIPS*, 2013.

Nitish Shirish Keskar and Richard Socher. Improving generalization performance by switching from adam to sgd. *arXiv preprint arXiv:1712.07628*, 2017.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *ICLR*, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

Paul Lévy and Paul Lévy. *Théorie de l'addition des variables aléatoires*. Gauthier-Villars, 1954.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *NeurIPS*, 2018.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *ICLR*, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *ICLR*, 2017.

Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. *ICLR*, 2019.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 1993.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *ICLR*, 2018.

Ilya Pavlyukevich. First exit times of solutions of stochastic differential equations driven by multiplicative lévy noise with heavy tails. *Stochastics and Dynamics*, 2011.

Martin Popel and Ondřej Bojar. Training tips for the transformer model. *PBML*, 2018.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

S Reddi, Manzil Zaheer, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. Adaptive methods for nonconvex optimization. In *NeurIPS*, 2018a.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *ICLR*, 2018b.

Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *IEEE international conference on neural networks*, 1993.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, 1951.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

Pedro Savarese, David McAllester, Sudarshan Babu, and Michael Maire. Domain-independent dominance of adaptive methods. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16286–16295, 2021.

Robin M Schmidt, Frank Schneider, and Philipp Hennig. Descending through a crowded valley–benchmarking deep learning optimizers. *ICML*, 2021.

Jun Shao. *Mathematical statistics*. Springer Science & Business Media, 2003.

Leon Simon et al. *Lectures on geometric measure theory*. The Australian National University, Mathematical Sciences Institute, Centre for Mathematics & its Applications, 1983.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.

Umut Simsekli, Levent Sagun, and Mert Gurbuzbalaban. A tail-index analysis of stochastic gradient noise in deep neural networks. In *ICML*, 2019.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *ICML*, 2013.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

Geoffrey Hinton Tijmen Tieleman. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *Coursera: Neural networks for machine learning*, 2012.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

Mengdi Wang, Ji Liu, and Ethan X Fang. Accelerating stochastic composition optimization. *Journal of Machine Learning Research*, 2017.

Frank Wilcoxon. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pp. 196–202. Springer, 1992.

Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nathan Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. *NeurIPS*, 2017.

Dmitry Yarotsky. Error bounds for approximations with deep relu networks. *Neural Networks*, 2017.

Han Zhang, Zizhao Zhang, Augustus Odena, and Honglak Lee. Consistency regularization for generative adversarial networks. In *ICLR*, 2020.

Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank J Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? *arXiv preprint arXiv:1912.03194*, 2019.

Pan Zhou, Jiashi Feng, Chao Ma, Caiming Xiong, Steven Hoi, and Weinan E. Towards theoretically understanding why sgd generalizes better than adam in deep learning. In *NeurIPS*, 2020.

Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *NeurIPS*, 2020.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, 2003.

A TECHNICAL DETAILS OF SUBSECTION 3.2

Here we provide more construction details and technical proofs for the Lévy-driven SDE in Adam-alike adaptive gradient algorithm equation 2. In the beginning we introduce a detailed derivation of the process equation 4 as well as its corresponding escaping set Υ in definition 3. Then we give some auxiliary theorems and lemmas, and summarize the proof of Lemma 1. Finally we prove the Lemma 1 and give a more detailed analysis of the conclusion that the expected escaping time of AdaM^3 is longer than that of Adam in a comparatively flat basin.

A.1 DERIVATION OF THE LÉVY-DRIVEN SDE EQUATION 4

To derive the SDE of Adam-alike algorithms equation 2, we firstly define $m_t' = \beta_1 m_{t-1}' + (1-\beta_1)\nabla f(\theta_t)$ with $m_0' = 0$. Then by the definition it holds that

$$m_t' - m_t = (\beta_1 - 1)\sum_{i=0}^{t} \beta_1^{t-i}\zeta_t.$$

Following Simsekli et al. (2019), the gradient noise ζ_t has heavy tails in reality and hence we assume that $\frac{1}{1-\beta_1^t}(m_t' - m_t)$ obeys $\mathcal{S}\widetilde{\alpha}\mathcal{S}$ distribution with time-dependent covariance matrix Σ_t. Since we can formulate equation 2 as

$$\theta_{t+1} = \theta_t - \alpha\frac{m_t'}{z_t} + \alpha\frac{(m_t' - m_t)}{z_t} \text{ where } z_t = (1-\beta_1^t)\sqrt{\frac{v_t}{(1-\beta_2^t)}}, \tag{5}$$

and we can replace the term $(m_t' - m_t)$ by $\alpha^{-\frac{1}{\widetilde{\alpha}}}(1-\beta_1^t)\Sigma_t S$ where each coordinate of S is independent and identically distributed as $\mathcal{S}\widetilde{\alpha}\mathcal{S}(1)$ based on the property of centered symmetric $\widetilde{\alpha}$-stable distribution. Let $R_t = \text{diag}(\sqrt{\frac{v_t}{(1-\beta_2^t)}})$, and we further assume that the step size α is small, then the continuous-time version of the process equation 5 becomes the following SDE:

$$d\theta_t = -R_t^{-1}\frac{m_t' dt}{(1-\beta_1^t)} + \alpha^{1-\frac{1}{\widetilde{\alpha}}}R_t^{-1}\Sigma_t dL_t,$$
$$dm_t = (1-\beta_1)(\nabla f(\theta_t) - m_t),\ dv_t = (1-\beta_2)(k_t^2 - v_t).$$

After replacing m_t' with m_t for brevity, we get the SDE equation 4 consequently.

A.2 PROOF OF LEMMA 1

To prove Lemma 1, we first introduce Theorem 2.

Theorem 2. Suppose Assumptions 1-3 hold. We define $\kappa_1 = \frac{c_1 L}{v_-|\tau_m - 1|}$ and $\kappa_2 = \frac{2\mu\tau}{\beta_1 v_+ + \mu\tau}\left(\beta_1 - \frac{\beta_2}{4}\right)$ with a constant c_1. Let $\upsilon^{\widetilde{\alpha}+1} = \Theta(\widetilde{\alpha})$, $\rho_0 = \frac{1}{16(1+c_2)}$ and $\ln\left(\frac{2\Delta}{\mu\upsilon^{1/3}}\right) \leq \kappa_2\upsilon^{-1/3}$ where $\Delta = f(\theta_0) - f(\theta^*)$ and a constant c_2. Then for any $\theta_0 \in \mathbf{\Omega}^{-2\upsilon^\gamma}$, $u > -1$, $\upsilon \in (0, \upsilon_0]$, $\gamma \in (0, \gamma_0]$ and $\rho \in (0, \rho_0]$ satisfying $\upsilon^\gamma \leq \rho_0$ and $\lim_{\upsilon\to 0}\rho = 0$, the Adam-alike algorithm in equation 2 obey

$$\frac{1-\rho}{1+u+\rho} \leq \mathbb{E}\left[\exp\left(-um(\Upsilon)\Theta(\upsilon^{-1})\Gamma\right)\right] \leq \frac{1+\rho}{1+u-\rho}.$$

From Theorem 2, by setting υ small, it holds that for any adaptive gradient algorithm the upper and lower bounds of its expected escaping time Γ is at the order of $\left(\frac{\upsilon}{m(\Upsilon)}\right)$, which directly implies Lemma 1 conclusively. Therefore, it suffices to validate Theorem 2.

The proof of Theorem 2 is given in Section A.2.3. Before we proceeed, we first provide some prerequisite notations in Section A.2.1 and list some useful theorems and lemmas in Section A.2.2.

A.2.1 PRELIMINARIES

For analyzing the uniform Lévy-driven SDEs in equation 4, we first introduce the Lévy process L_t into two components ξ_t and ε_i, namely

$$L_t = \xi_t + \varepsilon_t, \tag{6}$$

whose characteristic functions are respectively defined as

$$\mathbb{E}\left[e^{i\langle\lambda,\xi_t\rangle}\right] = e^{t\int_{\mathbb{R}^d\setminus\{0\}}\varepsilon I\left\{\|y\|_2\le\frac{1}{\upsilon^\delta}\right\}\nu(dy)},$$
$$\mathbb{E}\left[e^{i\langle\lambda,\varepsilon_t\rangle}\right] = e^{t\int_{\mathbb{R}^d\setminus\{0\}}\varepsilon I\left\{\|y\|_2\le\frac{1}{\upsilon^\delta}\right\}\nu(dy)},$$

where $\varepsilon = e^{i\langle\lambda,y\rangle} - 1 - i\langle\lambda,y\rangle I\left\{\|y\|_2 \le 1\right\}$ with υ defined in equation 4 and a constant δ s.t. $\upsilon^{-\delta} < 1$. Accordingly, the Lévy measure ν of the stochastic processes ξ and ε are

$$\nu_\xi = \nu\left(A \cap \left\{\|y\|_2 \le \frac{1}{\upsilon^\delta}\right\}\right), \quad \nu_\varepsilon = \nu\left(A \cap \left\{\|y\|_2 \ge \frac{1}{\upsilon^\delta}\right\}\right), \text{ where } A \in \mathcal{B}(\mathbb{R}^d).$$

Besides, for analysis, we should consider affects of the Lévy motion L_t to the Lévy-driven SDE of Adam variants. Here we define the Lévy-free SDE accordingly:

$$\begin{cases} d\widehat{\theta}_t = & -\mu_t \widehat{Q}_t^{-1}\widehat{m}_t, \\ d\widehat{m}_t = & (1-\beta_1)(\nabla f(\widehat{\theta}_t) - \widehat{m}_t), \\ d\widehat{v}_t = & (1-\beta_2)(\nabla(f\widehat{\theta}_t)^2 - \widehat{v}_t). \end{cases} \tag{7}$$

where $\widehat{Q}_t = \text{diag}(\sqrt{\widehat{v}_t})$.

A.2.2 Auxiliary theorems and lemmas

Theorem 3 (Adapted from Zhou et al. (2020))**.** Suppose Assumptions 1-3 hold. Assume the sequence $\{(\widehat{\theta}_t, \widehat{m}_t, \widehat{v}_t)\}$ are produced by equation 7. Let $\widehat{s}_t = \frac{h_t}{q_t}\left(\sqrt{\omega_t \widehat{v}_t}\right)$ with $h_t = 1-\beta_1$, $q_t = (1-(1-\beta_1)^t)^{-1}$ and $\omega_t = (1-(1-\beta_2)^t)^{-1}$. We define $\|x\|_y^2 = \sum_i y_i x_i^2$. Then for Lévy-driven Adam SDEs in equation 7, its Lyapunov function $\mathcal{L}(t) = f(\widehat{\theta}_t) - f(\widehat{\theta}^*) + \frac{1}{2}\|\widehat{m}_t\|_{\widehat{s}_t^{-1}}$ with the optimum solution θ^* in the current local basin $\mathbf{\Omega}$ obeys

$$\mathcal{L}(t) \le \Delta \exp\left(-\frac{2\mu\tau}{(1-\beta_1)v_+ + \mu\tau}\left(\frac{3}{4} - \beta_1 + \frac{\beta_2}{4}\right)t\right),$$

where $\Delta = f(\widehat{\theta}_0) - f(\widehat{\theta}^*)$ due to $\widehat{m}_0 = 0$. The sequence $\{\widehat{\theta}_t\}$ produced by equation 7 obeys

$$\left\|\widehat{\theta}_t - \theta^*\right\|_2^2 \le \frac{2\Delta}{\mu}\exp\left(-\frac{2\mu\tau}{(1-\beta_1)v_+ + \mu\tau}\left(\frac{3}{4} - \beta_1 + \frac{\beta_2}{4}\right)t\right).$$

Lemma 2 (Zhou et al. (2020))**.** (1) The process ξ in the Lévy process decomposition can be decomposed into two processes $\widehat{\xi}$ and linear drift, namely,

$$\xi_t = \widehat{\xi}_t + \mu_\upsilon t, \tag{8}$$

where $\widehat{\xi}$ is a zero mean Lévymartingale with bounded jumps.
(2) Let $\delta \in (0,1), \mu_\upsilon = \mathbb{E}(\xi_1)$ and $T_\upsilon = \upsilon^{-\theta}$ for some $\theta > 0, \rho_0 = \rho_0(\delta) = \frac{1-\delta}{4} > 0$ and $\theta_0 = \theta_0(\delta) = \frac{1-\delta}{3} > 0$. Suppose υ is sufficiently small such that $\Theta(1) \le \upsilon^{-\frac{1-\delta}{6}}$ and $\upsilon^{-\rho} - 2(C+\Theta(1))\upsilon^{\frac{7}{6}(1-\delta)+\frac{\rho}{2}} \ge 1$ with a constant $C = |\int_{0<u\le 1} u^2 d\Theta(u)| \in (0,+\infty)$. Then for all $\delta \in (0,\delta_0), \theta \in (0,\theta_0)$ there are $p_0 = p_0(\delta) = \frac{\delta}{2}$ and $\upsilon_0 = \upsilon_0(\delta,\rho)$ such that the estimates

$$\|\upsilon\xi_{T_\upsilon}\|_2 = \upsilon\|\mu_\upsilon\|_2 T_\upsilon < \upsilon^{2\rho} \text{ and } P([\upsilon\xi]_{T_\upsilon}^d \ge \upsilon^\rho) \le \exp(-\upsilon^{-p})$$

hold for all $p \in (0,p_0]$ and $\upsilon \in (0,\upsilon_0]$.

Lemma 3 (Zhou et al. (2020))**.** Let $\delta \in (0,1)$ and $g_{t\ge 0}^t$ be a bounded adapted càdlàg stochastic process with values in $\mathbb{R}^d$, $T_\upsilon = \upsilon^{-\theta}$, $\theta > 0$. Suppose $\sup_{t\ge 0}\|g^t\|$ is well bounded. Assume $\rho_0 = \rho_0(\delta) = \frac{1-\delta}{16} > 0$, $\theta_0 = \theta_0(\delta) = \frac{1-\delta}{3} > 0$, $p_0 = \frac{\rho}{2}$. For $\widehat{\xi}_t$ in equation 8, there is $\delta_0 = \delta_0(\delta) > 0$ such that for all $\rho \in (0,\rho_0)$ and $\theta \in (0,\theta_0)$, it holds

$$\mathbb{P}\left(\sup_{0\le t\le T_\upsilon}\upsilon\left|\sum_{i=1}^d\int_0^t g_{s-}^i d\widehat{\xi}_s^i\right| \ge \upsilon^\rho\right) \le 2\exp\left(-\upsilon^{-p}\right),$$

for all $p \in (0,p_0]$ and $0 < \upsilon \le \upsilon_0$ with $\upsilon_0 = \upsilon(\rho)$, where $\widehat{\xi}_s^i$ represents the i-th entry in $\widehat{\xi}_s$.

Lemma 4 (Zhou et al. (2020)). Under Assumptions 1-3 hold, assume $\delta \in (0,1), \rho_0 = \rho_0(\delta) = \frac{1-\delta}{16(1+c_1\kappa_1)} > 0, \theta_0 = \theta_0(\delta) = \frac{1-\delta}{3} > 0, p_0 = \min(\frac{\widehat{\rho}(1+c_1\kappa_1)}{2}, p), \frac{1}{c_2}\ln\left(\frac{2\Delta}{\mu\upsilon^{\widehat{\rho}}}\right) \leq \upsilon^{-\theta_0}$ where $\kappa_1 = \frac{c_2 l}{\upsilon_-|\tau_m - 1|}$ and $c_2 = \frac{2\mu\tau}{(1-\beta_1)\upsilon_+ + \mu\tau}\left(\frac{3}{4} - \beta_1 + \frac{\beta_2}{4}\right)$ in Adam-alike adaptive gradient algorithms. For all $\widehat{\rho} \in (0, \rho_0), p \in (0, p_0], 0 < \upsilon \leq \upsilon_0$ with $\upsilon_0 = \upsilon_0(\widehat{\rho})$, and $\theta_0 = \widehat{\theta}_0$, we have

$$\sup_{\theta_0 \in \mathbf{\Omega}} \mathbb{P}\left(\sup_{0 \leq t < \sigma_1} \left\|\theta_t - \widehat{\theta}_t\right\|_2 \geq 2\upsilon^{\widehat{\rho}}\right) \leq 2\exp(-\upsilon^{-\frac{p}{2}}), \tag{9}$$

where the sequences θ_t and $\widehat{\theta}_t$ are respectively produced by equation 4 and equation 7 in adaptive gradient method.

A.2.3 Proof of Theorem 2

Proof. The idea of this proof comes from equation 9 we showed in Lemma 4 where the sequence θ_t and $\widehat{\theta}_t$ start from the same initialization. Based on Theorem 3, we know that the sequence $\{\widehat{\theta}_t\}$ from equation 7 exponentially converges to the minimum θ^* of the local basin $\mathbf{\Omega}$. To escape the local basin $\mathbf{\Omega}$, we can either take small steps in the process ζ or large jumps J_k in the process ε. However, equation 9 suggests that these small jumps might not be helpful for escaping the basin. And for big jumps, the escaping time Γ of the sequence $\{\theta_t\}$ most likely occurs at the time σ_1 if the big jump υJ_1 in the process ε is large.
The verification of our desired results can be divided into two separate parts, namely establishing upper bound and lower bound of $\mathbb{E}\left[\exp\left(-um(\Upsilon)\Theta(\upsilon^{-1})\Gamma\right)\right]$ for any $u > -1$. Both of them can be established based on the following facts:

$$\begin{aligned}
&\left|\mathbb{P}\left(R_\theta^{-1}\Sigma_\theta \upsilon J_k \notin \mathbf{\Omega}^{\pm\upsilon^\gamma}, \|\upsilon J_k\|_2 \leq R\right) - \mathbb{P}\left(R_{\theta^*}^{-1}\Sigma_{\theta^*}\upsilon J_k \notin \mathbf{\Omega}^{\pm\upsilon^\gamma}, \|\upsilon J_k\|_2 \leq R\right)\right| \\
&\leq \frac{\delta'}{4}\cdot\frac{\Theta(\upsilon^{-1})}{\Theta(\upsilon^{-\delta})}, \\
&\left|\mathbb{P}\left(R_\theta^{-1}\Sigma_\theta \upsilon J_k \notin \mathbf{\Omega}, \|\upsilon J_k\|_2 \leq R\right) - \mathbb{P}\left(R_{\theta^*}^{-1}\Sigma_{\theta^*}\upsilon J_k \notin \mathbf{\Omega}, \|\upsilon J_k\|_2 \leq R\right)\right| \leq \frac{\delta'}{4}\cdot\frac{\Theta(\upsilon^{-1})}{\Theta(\upsilon^{-\delta})}, \\
&\mathbb{P}\left(R_{\theta^*}^{-1}\Sigma_{\theta^*}\upsilon J_k \notin \mathbf{\Omega}\right) - \mathbb{P}\left(R_{\theta^*}^{-1}\Sigma_{\theta^*}\upsilon J_k \notin \mathbf{\Omega}, \|\upsilon J_k\|_2 \leq R\right) \leq \frac{\delta'}{4}\cdot\frac{\Theta(\upsilon^{-1})}{\Theta(\upsilon^{-\delta})}.
\end{aligned} \tag{10}$$

Specifically, for the upper bound of $\mathbb{E}\left[\exp\left(-um(\Upsilon)\Theta(\upsilon^{-1})\Gamma\right)\right]$, we consider both the big jumps in the process ε and small jumps in the process ζ which may escape the local minimum. Instead of estimating the escaping time Γ from $\mathbf{\Omega}$, we first estimate the escaping time $\widetilde{\Xi}$ from $\mathbf{\Omega}^{-\bar{\rho}}$. Here we define the inner part of $\mathbf{\Omega}$ as $\mathbf{\Omega}^{-\bar{\rho}} := \{y \in \mathbf{\Omega} : \mathrm{dis}(\partial\mathbf{\Omega}, y) \geq \bar{\rho}\}$. Then by setting $\bar{\rho} \to 0$, we can use $\widetilde{\Xi}$ for a decent estimation of Γ. We denote $\bar{\rho} = \upsilon^\gamma$ where γ is a constant such that the results of Lemma 2-4 hold. So for the upper bound we mainly focus on $\widetilde{\Xi}$ in the beginning and then transfer the results to Γ. In the beginning, we can show that for any $u > -1$ it holds that,

$$\mathbb{E}\left[\exp\left(-um(\Upsilon)\Theta(\upsilon^{-1})\widetilde{\Xi}\right)\right] \leq \sum_{k=1}^{+\infty}\mathbb{E}\left[e^{-um(\Upsilon)\Theta(\upsilon^{-1})t_k} I\left\{\widetilde{\Xi} = t_k\right\} + Res_k\right],$$

where

$$Res_k \leq \begin{cases} \mathbb{E}\left[e^{-um(\Upsilon)\Theta(\upsilon^{-1})t_k} I\left\{\widetilde{\Xi} \in (t_{k-1}, t_k)\right\}\right], & \text{if } u \in (-1, 0] \\ \mathbb{E}\left[e^{-um(\Upsilon)\Theta(\upsilon^{-1})t_{k-1}} I\left\{\widetilde{\Xi} \in (t_{k-1}, t_k)\right\}\right], & \text{if } u \in (0, +\infty). \end{cases}$$

Then using the strong Markov property we can bound the first term $\mathbb{E}\left[e^{-um(\Upsilon)\Theta(\upsilon^{-1})t_k} I\left\{\widetilde{\Xi}=t_k\right\}\right]$ as

$$\begin{aligned} R_1=\sum_{k=1}^{+\infty}\mathbb{E}\left[e^{-um(\Upsilon)\Theta(\upsilon^{-1})t_k} I\left\{\Gamma=t_k\right\}\right] \leq & \frac{\alpha_\upsilon(1+\rho/3)}{1+u\alpha_\upsilon}\sum_{k=1}^{+\infty}\left(\frac{1-\alpha_\upsilon(1-\rho)}{1+u\alpha_\upsilon}\right)^{k-1} \\ \leq & \frac{\alpha_\upsilon(1+\rho/3)}{1+u\alpha_\upsilon}\sum_{k=0}^{+\infty}\left(\frac{1-\alpha_\upsilon(1-\rho)}{1+u\alpha_\upsilon}\right)^{k-1} \\ = & \frac{1+\rho/3}{1+u-\rho}. \end{aligned}$$

On the other hand, for the lower bound of $\mathbb{E}\left[\exp\left(-um(\Upsilon)\Theta(\upsilon^{-1})\Gamma\right)\right]$, we only consider the big jumps in the process ε which could escape from the basin, and ignore the probability that the small jumps in the process ζ which may also lead to an escape from the local minimum θ^*. Specifically, we can find a lower bound by discretization:

$$\mathbb{E}\left[\exp\left(-um(\Upsilon)\Theta(\upsilon^{-1})\Gamma\right)\right] \geq \sum_{k=1}^{+\infty}\mathbb{E}\left[\exp\left(-um(\Upsilon)\Theta(\upsilon^{-1})t_k\right)I\{\Gamma=t_k\}\right].$$

Then we can lower bound each term by three equations equation 10 we just listed here, which implies that for any $\theta_0 \in \boldsymbol{\Omega}^{-\upsilon^\gamma}$,

$$\mathbb{E}\left[e^{-um(\Upsilon)\Theta\upsilon^{-1}\Gamma}\right] \geq \frac{\alpha_\upsilon(1-\rho)}{1+u\alpha_\upsilon}\sum_{k=1}^{+\infty}\left(\frac{1-\alpha_\upsilon(1+\rho)}{1+u\alpha_\upsilon}\right)^{k-1} = \frac{1-\rho}{1+u+\rho},$$

where $\rho \to 0$ as $\upsilon \to 0$. The proof is completed. □

A.3 Proof of Proposition 1

Proof. Since we assumed the minimizer $\theta^* = \mathbf{0}$ in the basin Ω which is usually small,we can employ second-order Taylor expansion to approximate Ω as a quadratic basin whose center is θ^*. In other words, we can write

$$\Omega = \left\{ y \in \mathbb{R}^d \;\middle|\; f(\theta^*) + \frac{1}{2} y^\top H(\theta^*) y \leq h(\theta^*) \right\},$$

where $H(\theta^*)$ is the Hessian matrix at θ^* of function f and $h(\theta^*)$ is the basin height. Then according to Definition 3, we have

$$\Upsilon = \left\{ y \in \mathbb{R}^d \;\middle|\; y^\top \Sigma_{\theta^*} R_{\theta^*}^{-1} H(\theta^*) R_{\theta^*}^{-1} \Sigma_{\theta^*} y \geq h_f^* \right\}.$$

Here $R_{\theta^*} = \lim_{\theta_t \to \theta^*} \operatorname{diag}(\sqrt{v_t/(1-\beta_2^t)})$ is a matrix depending on the algorithm, $h_f^* = 2(h(\theta^*) - f(\theta^*))$ and Σ_{θ^*} is independent of the algorithm, i.e. the same for Adam and AdaM^3. Firstly, we will prove that $v_t^{(\text{AdaM}^3)} \geq v_t^{(\text{Adam})}$ when $t \to \infty$. To clarify the notation, we use θ_t, m_t, v_t, g_t to denote the symbols for Adam and $\widetilde{\theta}_t, \widetilde{m}_t, \widetilde{v}_t, \widetilde{g}_t$ for AdaM^3, and ζ_t is the gradient noise. By using Lemma 1 and above results, we have $\theta_t \approx \widetilde{\theta}_t \approx \theta^*$ before escaping when t is large, and thus $v_t = \lim_{\theta_t \to \theta^*} [\nabla f(\theta_t) + \zeta_t]^2$ and $\widetilde{v}_t = \lim_{\theta_t \to \theta^*} [\beta_1 \widetilde{m}_{t-1} + (1-\beta_1)(\nabla f(\widetilde{\theta}_t) + \zeta_t)]^2$. We will firstly show that $\mathbb{E}(\widetilde{v}_t) \geq \mathbb{E}(v_t)$ when t is large.

$$\begin{aligned} \mathbb{E}(v_t) &= \mathbb{E}(\lim_{\theta_t \to \theta^*} [\nabla f(\theta_t) + \zeta_t]^2) \overset{\text{(i)}}{=} \lim_{\theta_t \to \theta^*} \mathbb{E}([\nabla f(\theta_t) + \zeta_t]^2) \\ &= \lim_{\theta_t \to \theta^*} \left(\mathbb{E}(\nabla f(\theta_t)^2) + \mathbb{E}(2\nabla f(\theta_t)\zeta_t) + \mathbb{E}(\zeta_t^2)\right) \\ &\overset{\text{(ii)}}{=} \mathbb{E}(\lim_{\theta_t \to \theta^*} \nabla f(\theta_t)^2) + \lim_{\theta_t \to \theta^*} \mathbb{E}(2\nabla f(\theta_t)\zeta_t) + \lim_{\theta_t \to \theta^*} \mathbb{E}(\zeta_t^2) \\ &\overset{\text{(iii)}}{=} \lim_{\theta_t \to \theta^*} \mathbb{E}(\zeta_t^2), \end{aligned}$$

where (i) and (ii) are due to the dominated convergence theorem (DCT) since we have that we know both $\|\nabla f(\theta_t)\|_2$ and $\|\nabla f(\theta_t)+\zeta_t\|_2$ could be bounded by H in Assumption 4. And (iii) is due to the fact that $\nabla f(\theta^*)=0$ since function f attains its minimum point at θ^*, and ζ_t has zero mean, i.e.

$$\lim_{\theta_t\to\theta^*}\mathbb{E}(\nabla f(\theta_t)\zeta_t)=\lim_{\theta_t\to\theta^*}\mathbb{E}(\nabla f(\theta_t))\mathbb{E}(\zeta_t)=0.$$

And similarly we can prove that,

$$\begin{aligned}\mathbb{E}(\widetilde{v}_t)&=\mathbb{E}\left(\lim_{\theta_t\to\theta^*}[\beta_1\widetilde{m}_{t-1}+(1-\beta_1)(\nabla f(\widetilde{\theta}_t)+\zeta_t)]^2\right)\\&=\lim_{\theta_t\to\theta^*}\left(\mathbb{E}(\beta_1^2\widetilde{m}_{t-1}^2)+\mathbb{E}((1-\beta_1)^2(\nabla f(\widetilde{\theta}_t)+\zeta_t)^2)+\mathbb{E}(2\beta_1(1-\beta_1)\widetilde{m}_{t-1}\nabla(f(\widetilde{\theta}_t)+\zeta_t))\right)\\&\overset{\text{(i)}}{=}\beta_1^2\lim_{\theta_t\to\theta^*}\mathbb{E}(\widetilde{m}_{t-1}^2)+(1-\beta_1)^2\lim_{\theta_t\to\theta^*}\mathbb{E}(\zeta_t^2),\end{aligned}$$

where we can get the equality (i) simply by the same argument with dominated convergence theorem we just used:

$$\begin{aligned}\lim_{\widetilde{\theta}_t\to\theta^*}\mathbb{E}(\nabla(f(\widetilde{\theta}_t)^2)&=\mathbb{E}(\lim_{\widetilde{\theta}_t\to\theta^*}\nabla(f(\widetilde{\theta}_t)^2)\overset{\text{(i)}}{=}0,\\\lim_{\widetilde{\theta}_t\to\theta^*}\mathbb{E}(\nabla(f(\widetilde{\theta}_t)\zeta_t)&=\mathbb{E}(\lim_{\widetilde{\theta}_t\to\theta^*}\nabla(f(\widetilde{\theta}_t)\zeta_t)\overset{\text{(ii)}}{=}0,\\\lim_{\widetilde{\theta}_t\to\theta^*}\mathbb{E}(\widetilde{m}_{t-1}(\nabla f(\widetilde{\theta}_t)+\zeta_t))&=\mathbb{E}(\lim_{\widetilde{\theta}_t\to\theta^*}\widetilde{m}_{t-1}\nabla f(\widetilde{\theta}_t))+\lim_{\widetilde{\theta}_t\to\theta^*}\mathbb{E}(\widetilde{m}_{t-1})\mathbb{E}(\zeta_t)\overset{\text{(iii)}}{=}0,\end{aligned}$$

where we get the equality (i) and (ii) since the function $f(\widetilde{\theta}_t)^2$ and $f(\widetilde{\theta}_t)\zeta_t$ could be absolutely bounded by H^2. And the first term in equality (iii) is 0 since we have $\|\widetilde{m}_{t-1}\|_2\le H$ by its definition and $\nabla f(\theta^*)=0$, and the second term vanishes since the noise ζ_t has zero mean. Based on the Assumption 5, we have

$$\mathbb{E}(\widetilde{m}_{t-1}^2)\ge\frac{2-\beta_1}{\beta_1}\mathbb{E}(\zeta_t^2),$$

which implies that $\mathbb{E}(\widetilde{v}_t)\ge\mathbb{E}(v_t)$ when t is large. It further indicates that $R_{\theta^*}^{(\text{ADAM}^3)}\ge R_{\theta^*}^{(\text{ADAM})}$. We consider the volume of the complementary set

$$\Upsilon^c=\left\{y\in\mathbb{R}^d\;\middle|\;y^\top\Sigma_{\theta^*}R_{\theta^*}^{-1}H(\theta^*)R_{\theta^*}^{-1}\Sigma_{\theta^*}y<h_f^*\right\},$$

which can be viewed as a d-dimensional ellipsoid. We can further decompose the symmetric matrix $M\coloneqq\Sigma_{\theta^*}R_{\theta^*}^{-1}H(\theta^*)R_{\theta^*}^{-1}\Sigma_{\theta^*}$ by SVD decomposition

$$M=U^\top AU,$$

where U is an orthogonal matrix and A is a diagonal matrix with nonnegative elements. Hence the transformation $y\to Uy$ is an orthogonal transformation which means the volume of Υ^c equals the volume of set

$$\left\{y'\in\mathbb{R}^d\;\middle|\;y'^\top Ay'<h_f^*\right\}.$$

Considering the fact that the volume of a d-dimensional ellipsoid centered at $\mathbf{0}$ $E_d(r)=\{(x_1,x_2,\cdots,x_n):\sum_{i=1}^d\frac{x_i^2}{R_i^2}\le1\}$ is

$$V(E_d(r))=\frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2}+1)}\Pi_{i=1}^nR_i,$$

and the fact we just proved that $R_{\theta^*}^{(\text{ADAM}^3)}\ge R_{\theta^*}^{(\text{ADAM})}$. Therefore we deduce the volume of $\Upsilon^{(\text{ADAM}^3)}$ is smaller than that of $\Upsilon^{(\text{ADAM})}$, which indicates that for Radon measure $m(\cdot)$ we have $m(\Upsilon^{(\text{ADAM}^3)})\ge m(\Upsilon^{(\text{ADAM})})$. Based on Lemma 1, we consequently have $\mathbb{E}(\Gamma^{(\text{ADAM}^3)})\ge\mathbb{E}(\Gamma^{(\text{ADAM})})$. □

B PROOFS IN SECTION 4

B.1 CONVERGENCE ANALYSIS IN CONVEX OPTIMIZATION

We analyze the convergence of AdaM^3 in convex setting utilizing the online learning framework (Zinkevich, 2003). Given a sequence of convex cost functions $f_1(\theta), \cdots, f_T(\theta)$, the regret is defined as $R(T) = \sum_{t=1}^{T}[f_t(\theta_t) - f_t(\theta^*)]$, where $\theta^* = \arg\min_\theta \sum_{t=1}^{T} f_t(\theta)$ is the optimal parameter and f_t can be interpreted as the loss function at the t-th step. Then we have:

Theorem 4. Let $\{\theta_t\}$ and $\{v_t\}$ be the sequences yielded by AdaM^3. Let $\alpha_t = \alpha/\sqrt{t}, \beta_{1,1} = \beta_1, 0 < \beta_{1,t} \leq \beta_1 < 1, v_t \leq v_{t+1}$ for all $t \in [T]$ and $\gamma = \beta_1/\sqrt{\beta_2} < 1$. Assume that the distance between any θ_t generated by AdaM^3 is bounded, $\|\theta_m - \theta_n\|_\infty \leq D_\infty$ for any $m, n \in \{1, \cdots, T\}$. Then we have the following bound:

$$R(T) \leq \frac{D_\infty^2 \sqrt{T}}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{v_{T,i}} + \frac{D_\infty^2}{2(1-\beta_1)} \sum_{t=1}^{T} \sum_{i=1}^{d} \frac{\beta_{1,t}\sqrt{v_{t,i}}}{\alpha_t} + \frac{\alpha\sqrt{1+\log T}}{(1-\beta_1)^3(1-\gamma)\sqrt{1-\beta_2}} \sum_{i=1}^{d} \|g_{1:T,i}\|_2 .$$

Theorem 4 implies that the regret of AdaM^3 can be bounded by $\widetilde{O}^3(\sqrt{T})$, especially when the data features are sparse as Section 1.3 in Duchi et al. (2011) and then we have $\sum_{i=1}^{d} \sqrt{v_{T,i}} \ll \sqrt{d}$ and $\sum_{i=1}^{d} \|g_{1:T,i}\|_2 \ll \sqrt{dT}$. Imposing additional assumptions that $\beta_{1,t}$ decays exponentially and that the gradients of f_t are bounded (Kingma & Ba, 2015; Liu et al., 2019), we can obtain:

Corollary 2. Further Suppose $\beta_{1,t} = \beta_1 \lambda^t$ and the function f_t has bounded gradients, $\|\nabla f_t(\theta)\|_\infty \leq G_\infty$ for all $\theta \in \mathbb{R}^d$, AdaM^3 achieves the guarantee $R(T)/T = \widetilde{O}(1/\sqrt{T})$ for all $T \geq 1$:

$$\frac{R(T)}{T} \leq \left[\frac{d\alpha\sqrt{1+\log T}}{(1-\beta_1)^3(1-\gamma)\sqrt{(1-\beta_2)T}} + \frac{dD_\infty^2}{2\alpha(1-\beta_1)\sqrt{T}} \right] \cdot (G_\infty + \sqrt{\epsilon/1-\beta_2}) + \frac{dD_\infty^2 G_\infty \beta_1}{2\alpha(1-\beta_1)(1-\lambda)^2 T}.$$

From Corollary 2, the average regret of AdaM^3 converges to zero as T goes to infinity. The proofs of Theorem 4 and Corollary 2 are provided in Appendix B.2.

B.2 PROOF OF THE CONVERGENCE RESULTS FOR THE CONVEX CASE

[3] $\widetilde{O}(\cdot)$ denotes $O(\cdot)$ with hidden logarithmic factors.

B.2.1 PROOF OF THEOREM 4

Proof. Firstly, according to the definition of AdaM^3 in Algorithm 1, by algebraic shrinking we have

$$
\begin{aligned}
\sum_{t=1}^{T}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} &= \sum_{t=1}^{T-1}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} + \frac{\left(\sum_{j=1}^{T}(1-\beta_{1,j})\Pi_{k=1}^{T-j}\beta_{1,T-k+1}g_{j,i}\right)^2}{\sqrt{T\left[\sum_{j=1}^{T}(1-\beta_2)\beta_2^{T-j}m_{j,i}^2+\epsilon+\sum_{j=1}^{T-1}\prod_{i=1}^{j}\beta_2^i\epsilon\right]}} \\
&\leq \sum_{t=1}^{T-1}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} + \frac{\left(\sum_{j=1}^{T}(1-\beta_{1,j})\Pi_{k=1}^{T-j}\beta_{1,T-k+1}g_{j,i}\right)^2}{\sqrt{T\sum_{j=1}^{T}(1-\beta_2)\beta_2^{T-j}m_{j,i}^2}} \\
&\leq \sum_{t=1}^{T-1}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} + \frac{(\sum_{j=1}^{T}\Pi_{k=1}^{T-j}\beta_{1,T-k+1})(\sum_{j=1}^{T}\Pi_{k=1}^{T-j}\beta_{1,T-k+1}g_{j,i}^2)}{\sqrt{T\sum_{j=1}^{T}(1-\beta_2)\beta_2^{T-j}m_{j,i}^2}} \\
&\overset{\text{(i)}}{\leq} \sum_{t=1}^{T-1}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} + \frac{(\sum_{j=1}^{T}\beta_1^{T-j})(\sum_{j=1}^{T}\beta_1^{T-j}g_{j,i}^2)}{\sqrt{T(1-\beta_2)\sum_{j=1}^{T}\beta_2^{T-j}m_{j,i}^2}} \\
&\leq \sum_{t=1}^{T-1}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} + \frac{1}{1-\beta_1}\frac{\sum_{j=1}^{T}\beta_1^{T-j}g_{j,i}^2}{\sqrt{T(1-\beta_2)\sum_{j=1}^{T}\beta_2^{T-j}m_{j,i}^2}} \\
&= \sum_{t=1}^{T-1}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} \\
&\quad + \frac{1}{(1-\beta_1)\sqrt{T(1-\beta_2)}}\sum_{j=1}^{T}\frac{\beta_1^{T-j}g_{j,i}^2}{\sqrt{\sum_{j=1}^{T}\beta_2^{T-j}\left(\sum_{l=1}^{j}(1-\beta_{1,l})\Pi_{k=1}^{j-l}\beta_{1,j-k+1}g_{l,i}\right)^2}} \\
&\leq \sum_{t=1}^{T-1}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} + \frac{1}{(1-\beta_1)\sqrt{T(1-\beta_2)}}\sum_{j=1}^{T}\frac{\beta_1^{T-j}g_{j,i}^2}{\sqrt{\sum_{j=1}^{T}\beta_2^{T-j}\left((1-\beta_{1,j})g_{j,l}\right)^2}} \\
&\leq \sum_{t=1}^{T-1}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} + \frac{1}{(1-\beta_1)\sqrt{T(1-\beta_2)}}\sum_{j=1}^{T}\frac{\beta_1^{T-j}g_{j,i}^2}{\sqrt{\beta_2^{T-j}(1-\beta_{1,j})^2g_{j,i}^2}} \\
&\overset{\text{(ii)}}{\leq} \sum_{t=1}^{T-1}\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} + \frac{1}{(1-\beta_1)^2\sqrt{T(1-\beta_2)}}\sum_{j=1}^{T}\gamma^{T-j}g_{j,i},
\end{aligned}
$$

where (i) arises from $\beta_{1,t} \leq \beta_1$, and (ii) comes from the definition that $\gamma = \frac{\beta_1}{\sqrt{\beta_2}}$. Then by induction, we have

$$\begin{aligned}
\sum_{t=1}^{T} \frac{m_{t,i}^2}{\sqrt{t v_{t,i}}} &\leq \sum_{t=1}^{T} \frac{1}{(1-\beta_1)^2 \sqrt{t(1-\beta_2)}} \sum_{j=1}^{t} \gamma^{t-j} g_{j,i} \\
&\leq \frac{1}{(1-\beta_1)^2 \sqrt{1-\beta_2}} \sum_{t=1}^{T} \frac{1}{\sqrt{t}} \sum_{j=1}^{t} \gamma^{t-j} g_{j,i} \\
&\overset{(i)}{\leq} \frac{1}{(1-\beta_1)^2 \sqrt{1-\beta_2}} \sum_{t=1}^{T} g_{t,i} \sum_{j=t}^{T} \frac{\gamma^{j-t}}{\sqrt{j}} \\
&\leq \frac{1}{(1-\beta_1)^2 \sqrt{1-\beta_2}} \sum_{t=1}^{T} g_{t,i} \cdot \frac{1}{(1-\gamma)\sqrt{t}} \\
&\leq \frac{1}{(1-\beta_1)^2 (1-\gamma) \sqrt{1-\beta_2}} \sum_{t=1}^{T} \frac{g_{t,i}}{\sqrt{t}} \\
&\overset{(ii)}{\leq} \frac{1}{(1-\beta_1)^2 (1-\gamma) \sqrt{1-\beta_2}} \left\| g_{1:T,i} \right\|_2 \sqrt{\sum_{t=1}^{T} \frac{1}{t}} \\
&\overset{(iii)}{\leq} \frac{\sqrt{1+\log T}}{(1-\beta_1)^2 (1-\gamma) \sqrt{1-\beta_2}} \left\| g_{1:T,i} \right\|_2 ,
\end{aligned}$$

where (i) exchanges the indices of summing, (ii) employs Cauchy-Schwarz Inequality and (iii) comes from the following bound on harmonic sum:

$$\sum_{t=1}^{T} \frac{1}{t} \leq 1 + \log T.$$

Due to convexity of f_t, we get

$$\begin{aligned}
f_t(\theta_t) - f_t(\theta^*) &\leq g_t^\top (\theta_t - \theta^*) \\
&= \sum_{i=1}^{d} g_{t,i} (\theta_{t,i} - \theta_{,i}^*). \qquad (11)
\end{aligned}$$

According to the updating rule, we have

$$\begin{aligned}
\theta_{t+1} &= \theta_t - \alpha_t \frac{m_t}{\sqrt{v_t}} \\
&= \theta_t - \alpha_t \left(\frac{\beta_{1,t}}{\sqrt{v_t}} m_{t-1} + \frac{1-\beta_{1,t}}{\sqrt{v_t}} g_t \right). \qquad (12)
\end{aligned}$$

Substracting θ^*, squaring both sides and considering only the i-th element in vectors, we obtain

$$(\theta_{t+1,i} - \theta_{,i}^*)^2 = (\theta_{t,i} - \theta_{,i}^*)^2 - 2\alpha_t \left(\frac{\beta_{1,t}}{\sqrt{v_{t,i}}} m_{t-1,i} + \frac{1-\beta_{1,t}}{\sqrt{v_{t,i}}} g_{t,i} \right) (\theta_{t,i} - \theta_{,i}^*) + \alpha_t^2 \left(\frac{m_{t,i}}{\sqrt{v_{t,i}}} \right)^2 .$$

By rearranging the terms, we have

$$\begin{aligned}
&2\alpha_t \frac{1-\beta_{1,t}}{\sqrt{v_{t,i}}} g_{t,i} (\theta_{t,i} - \theta_{,i}^*) \\
&= (\theta_{t,i} - \theta_{,i}^*)^2 - (\theta_{t+1,i} - \theta_{,i}^*)^2 - 2\alpha_t \cdot \frac{\beta_{1,t}}{\sqrt{v_{t,i}}} \cdot m_{t-1,i} (\theta_{t,i} - \theta_{,i}^*) + \alpha_t^2 \left(\frac{m_{t,i}}{\sqrt{v_{t,i}}} \right)^2 .
\end{aligned}$$

Further we have

$$
\begin{aligned}
g_{t,i}(\theta_{t,i}-\theta^*_{,i}) =& \frac{\sqrt{v_{t,i}}}{2\alpha_t(1-\beta_{1,t})}[(\theta_{t,i}-\theta^*_{,i})^2-(\theta_{t+1,i}-\theta^*_{,i})^2]+\frac{\alpha_t\sqrt{v_{t,i}}}{2(1-\beta_{1,t})}\left(\frac{m_{t,i}}{\sqrt{v_{t,i}}}\right)^2 \\
&+\frac{\beta_{1,t}}{1-\beta_{1,t}}(\theta^*_{,i}-\theta_{t,i})m_{t-1,i} \\
=& \frac{\sqrt{v_{t,i}}}{2\alpha_t(1-\beta_{1,t})}[(\theta_{t,i}-\theta^*_{,i})^2-(\theta_{t+1,i}-\theta^*_{,i})^2]+\frac{\alpha_t\sqrt{v_{t,i}}}{2(1-\beta_{1,t})}\left(\frac{m_{t,i}}{\sqrt{v_{t,i}}}\right)^2 \\
&+\frac{\beta_{1,t}}{1-\beta_{1,t}}\cdot\frac{v_{t,i}^{\frac{1}{4}}}{\sqrt{\alpha_t}}\cdot(\theta^*_{,i}-\theta_{t,i})\cdot\sqrt{\alpha_t}\cdot\frac{m_{t-1,i}}{v_{t,i}^{\frac{1}{4}}} \\
\le& \frac{\sqrt{v_{t,i}}}{2\alpha_t(1-\beta_1)}[(\theta_{t,i}-\theta^*_{,i})^2-(\theta_{t+1,i}-\theta^*_{,i})^2]+\frac{\alpha}{2(1-\beta_1)}\cdot\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} \qquad (13) \\
&+\frac{\beta_{1,t}}{2\alpha_t(1-\beta_{1,t})}(\theta^*_{,i}-\theta_{t,i})^2\sqrt{v_{t,i}}+\frac{\beta_1\alpha}{2(1-\beta_1)}\cdot\frac{m_{t-1,i}^2}{\sqrt{tv_{t,i}}}, \qquad (14)
\end{aligned}
$$

where equation 14 bounds the last term of equation 13 by Cauchy-Schwarz Inequality and plugs in the value of α_t. Plugging equation 14 into equation 12 and summing from $t=1$ to T, we obtain

$$
\begin{aligned}
R(T) =& \sum_{t=1}^{T}\sum_{i=1}^{d} g_{t,i}(\theta_{t,i}-\theta^*_{,i}) \\
\le& \sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\sqrt{v_{t,i}}}{2\alpha_t(1-\beta_1)}[(\theta_{t,i}-\theta^*_{,i})^2-(\theta_{t+1,i}-\theta^*_{,i})^2]+\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\alpha}{2(1-\beta_1)}\cdot\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}} \\
&+\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_{1,t}}{2\alpha_t(1-\beta_{1,t})}(\theta^*_{,i}-\theta_{t,i})^2\sqrt{v_{t,i}}+\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_1\alpha}{2(1-\beta_1)}\cdot\frac{m_{t-1,i}^2}{\sqrt{tv_{t,i}}} \\
\le& \sum_{i=1}^{d}\frac{\sqrt{v_{1,i}}}{2\alpha_1(1-\beta_1)}(\theta_{1,i}-\theta^*_{,i})^2+\frac{1}{2(1-\beta_1)}\sum_{t=2}^{T}\sum_{i=1}^{d}(\theta_{t,i}-\theta^*_{,i})^2\left(\frac{\sqrt{v_{t,i}}}{\alpha_t}-\frac{\sqrt{v_{t-1,i}}}{\alpha_{t-1}}\right) \qquad (15) \\
&+\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_{1,t}}{2\alpha_t(1-\beta_1)}(\theta^*_{,i}-\theta_{t,i})^2\sqrt{v_{t,i}}+\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\alpha}{1-\beta_1}\cdot\frac{m_{t,i}^2}{\sqrt{tv_{t,i}}}, \qquad (16)
\end{aligned}
$$

where equation 16 rearranges the first term of equation 15. Finally utilizing the assumptions in Theorem 4, we get

$$
\begin{aligned}
R(T) \le& \sum_{i=1}^{d}\frac{\sqrt{v_{1,i}}}{2\alpha_1(1-\beta_1)}D_\infty^2+\frac{1}{2(1-\beta_1)}\sum_{t=2}^{T}\sum_{i=1}^{d}D_\infty^2\left(\frac{\sqrt{v_{t,i}}}{\alpha_t}-\frac{\sqrt{v_{t-1,i}}}{\alpha_{t-1}}\right) \\
&+\frac{D_\infty^2}{2(1-\beta_1)}\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_{1,t}v_{t,i}^{\frac{1}{2}}}{\alpha_t}+\sum_{i=1}^{d}\frac{\alpha\sqrt{1+\log T}}{(1-\beta_1)^3(1-\gamma)\sqrt{1-\beta_2}}\left\|g_{1:T,i}\right\|_2 \\
=& \sum_{i=1}^{d}\frac{\sqrt{v_{T,i}}}{2\alpha_T(1-\beta_1)}D_\infty^2+\frac{D_\infty^2}{2(1-\beta_1)}\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_{1,t}v_{t,i}^{\frac{1}{2}}}{\alpha_t} \\
&+\sum_{i=1}^{d}\frac{\alpha\sqrt{1+\log T}}{(1-\beta_1)^3(1-\gamma)\sqrt{1-\beta_2}}\left\|g_{1:T,i}\right\|_2, \qquad (17)
\end{aligned}
$$

which is our desired result. □

B.2.2 PROOF OF COROLLARY 2

Proof. Plugging $\alpha_t = \frac{\alpha}{\sqrt{t}}$ and $\beta_{1,t} = \beta_1 \lambda^t$ into equation 17, we get

$$R(T) \leq \frac{D_\infty^2 \sqrt{T}}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{v_{T,i}} + \frac{D_\infty^2}{2\alpha(1-\beta_1)} \sum_{t=1}^{T} \sum_{i=1}^{d} \beta_1 \lambda^t \sqrt{t v_{t,i}} + \sum_{i=1}^{d} \frac{\alpha\sqrt{1+\log T}}{(1-\beta_1)^3(1-\gamma)\sqrt{1-\beta_2}} \left\| g_{1:T,i} \right\|_2 . \tag{18}$$

Next, we employ Mathematical Induction to prove that $v_t, i \leq G_\infty$ for any $0 \leq t \leq T, 1 \leq i \leq d$. $\forall i$, we have $m_{0,i}^2 = 0 \leq G_\infty^2$. Suppose $m_{t-1,i} \leq G_\infty$, we have

$$\begin{aligned} m_{t,i}^2 &= (\beta_{1,t} m_{t-1,i} + (1-\beta_{1,t}) g_{t,i})^2 \\ &\overset{\text{(i)}}{\leq} \beta_{1,t} m_{t-1,i}^2 + (1-\beta_{1,t}) g_{t,i}^2 \\ &\leq \beta_{1,t} G_\infty^2 + (1-\beta_{1,t}) G_\infty^2 = G_\infty^2, \end{aligned}$$

where (i) comes from the convexity of function $f = x^2$. Hence by induction, we have $m_{t,i}^2 \leq G_\infty^2$ for all $0 \leq t \leq T$. Furthermore, $\forall i$, we have $v_{0,i} = 0 \leq G_\infty^2$. Suppose $v_{t-1,i} \leq G_\infty^2 + (1 - \beta_2^{t-1})\epsilon/(1-\beta_2)$, we have

$$\begin{aligned} v_{t,i} &= \beta_2 v_{t-1,i} + (1-\beta_2) m_{t,i}^2 + \epsilon \\ &\leq \beta_2 G_\infty^2 + (1-\beta_2) G_\infty^2 + \left(\frac{\beta_2 - \beta_2^t}{1-\beta_2} + 1 \right) \epsilon = G_\infty^2 + \frac{1-\beta_2^t}{1-\beta_2} \epsilon. \end{aligned}$$

Therefore, by induction, we have $v_{t,i} \leq G_\infty^2 + (1-\beta_2^t)\epsilon/(1-\beta_2) \leq G_\infty^2 + \epsilon/(1-\beta_2), \forall i, t$. Combining this with the fact that $\sum_{i=1}^{d} \left\| g_{1:T,i} \right\|_2 \leq d G_\infty \sqrt{T}$ and equation 18, we obtain

$$\begin{aligned} R(T) \leq & \frac{d(G_\infty + \sqrt{\frac{\epsilon}{1-\beta_2}}) D_\infty^2 \sqrt{T}}{2\alpha(1-\beta_1)} \\ & + \frac{d(G_\infty + \sqrt{\frac{\epsilon}{1-\beta_2}}) D_\infty^2 \beta_1}{2\alpha(1-\beta_1)} \sum_{t=1}^{T} \lambda^t \sqrt{t} + \frac{d G_\infty \alpha \sqrt{1+\log T}}{(1-\beta_1)^3(1-\gamma)\sqrt{(1-\beta_2)T}}. \end{aligned} \tag{19}$$

For $\sum_{t=1}^{T} \lambda^t \sqrt{t}$, we apply arithmetic geometric series upper bound:

$$\sum_{t=1}^{T} \lambda^t \sqrt{t} \leq \sum_{t=1}^{T} t \lambda^t \leq \frac{1}{(1-\lambda)^2}. \tag{20}$$

Plugging equation 20 into equation 19 and dividing both sides by T, we obtain

$$\frac{R(T)}{T} \leq \frac{d(G_\infty + \sqrt{\frac{\epsilon}{1-\beta_2}})\alpha\sqrt{1+\log T}}{(1-\beta_1)^3(1-\gamma)\sqrt{(1-\beta_2)T}} + \frac{d D_\infty^2 (G_\infty + \sqrt{\frac{\epsilon}{1-\beta_2}})}{2\alpha(1-\beta_1)\sqrt{T}} + \frac{d D_\infty^2 G_\infty \beta_1}{2\alpha(1-\beta_1)(1-\lambda)^2 T},$$

which concludes the proof. □

B.3 PROOF OF THE CONVERGENCE RESULTS FOR THE NON-CONVEX CASE

B.3.1 USEFUL LEMMA

Lemma 5. (Wang et al. (2017); Guo et al. (2021)) Consider a moving average sequence $m_{t+1} = \beta_{1,t} m_t + (1-\beta_t) g_{t+1}$ for tracking $\nabla f(\theta_t)$, where $\mathbb{E}(g_{t+1}) = \nabla f(\theta_t)$ and f is an L-Lipschits continuous mapping. Then we have

$$\begin{aligned} \mathbb{E}_t(\left\| m_{t+1} - \nabla f(\theta_t) \right\|_2^2) \leq & \beta_{1,t} \left\| m_t - \nabla f(\theta_{t-1}) \right\|_2^2 + 2(1-\beta_{1,t})^2 \mathbb{E}_t(\left\| g_{t+1} - \nabla f(\theta_t) \right\|_2^2) \\ & + \frac{L^2}{1-\beta_{1,t}} \left\| \theta_t - \theta_{t-1} \right\|_2^2 . \end{aligned}$$

Based on the above Lemma 5, we could derive the following convergence result in Theorem 1.

B.3.2 PROOF OF THEOREM 1

We denote $\Delta_t = \|m_{t+1} - \nabla f(\theta_2)\|_2^2$, and by applying Lemma 5 we can get:

$$\mathbb{E}_t(\Delta_{t+1}) \leq \beta_{1,t+1}\Delta_t + 2(1-\beta_{1,t+1})^2\mathbb{E}_t(\|g_{t+2} - \nabla f(\theta_{t+1})\|_2^2) + \frac{L^2}{1-\beta_{1,t+1}}\|\theta_{t+1} - \theta_t\|_2^2. \tag{21}$$

Based on some simple calculation, we can verify that $\sum_{i=0}^{t-1}\beta_2^i\epsilon = (1-\beta_2^t)\epsilon/(1-\beta_2)$, which implies that $1/\sqrt{v_t} \leq b_{u,t}$ holds for all $t \in [T]$ elementwisely. On the other hand, since we have $m_{t+1} = \beta_{1,t}m_{t-1} + (1-\beta_{1,t})g_t$ with the condition $\|g_t\|_\infty \leq G$ for all $t \in [T]$. Therefore, we can deduce that

$$\|m_t\|_\infty \leq \beta_{1,t}\|m_{t-1}\|_\infty + (1-\beta_{1,t})G \leq \beta\|m_{t-1}\|_\infty + (1-\beta)G, \quad m_0 = 0,$$

which implies that $\|m_t\|_\infty \leq G(1-\beta^t)$ after some simple calculation, and hence we have $m_t^2 \leq G^2(1-\beta^T)^2$ elementwise. Next, since we have $v_{t+1} = \beta_2 v_{t-1} + (1-\beta_2)m_t^2 + \epsilon$, we can similarly get

$$\|v_t\|_\infty \leq \beta_2\|v_{t-1}\|_\infty + (1-\beta_2)\left(G^2(1-\beta_{1,1}^T) + \frac{\epsilon}{1-\beta_2}\right), \quad v_0 = 0,$$

which implies that $\|v_t\|_\infty \leq \left(G^2(1-\beta^T) + \frac{\epsilon}{1-\beta_2}\right)(1-\beta_2^t)$ and hence $1/\sqrt{v_t} \geq b_{l,t}$. After some simplification of equation 21, we have

$$\begin{aligned}
&\mathbb{E}_t(\sum_{t=0}^{T}(1-\beta_{1,t+1})\Delta_t) \\
&\leq \mathbb{E}\left[\sum_{t=0}^{T}(\Delta_t - \Delta_{t-1}) + \sum_{t=0}^{T}2\sigma^2(1-\beta_{1,t+1})^2 + \sum_{t=0}^{T}\frac{L^2}{1-\beta_{1,t+1}}\|\theta_{t+1} - \theta_t\|_2^2\right] \\
&\overset{\text{(i)}}{=} \mathbb{E}\left[\sum_{t=0}^{T}(\Delta_t - \Delta_{t-1}) + \sum_{t=0}^{T}2\sigma^2(1-\beta_{1,t+1})^2 + \sum_{t=0}^{T}\frac{L^2\alpha_t^2 b_{u,t+1}^2}{1-\beta_{1,t+1}}\|m_{t+1}\|_2^2\right],
\end{aligned} \tag{22}$$

where (i) comes from the Lipschitz property of ∇f. On the other hand, since f has Lipschitz gradient, we have:

$$\begin{aligned}
f(\theta_{t+1}) \leq& f(\theta_t) + \nabla f(\theta_t)^\top(\theta_{t+1} - \theta_t) + \frac{L}{2}\|\theta_{t+1} - \theta_t\|_2^2 \\
=& f(\theta_t) - \nabla f(\theta_t)^\top(\frac{\alpha_t}{\sqrt{v_t}}m_{t+1}) + \frac{L}{2}\left\|\frac{\alpha_t}{\sqrt{v_t}}m_{t+1}\right\|_2^2 \\
=& f(\theta_t) + \frac{\alpha_t}{2\sqrt{v_t}}\|\nabla f(\theta_t) - m_{t+1}\|_2^2 + \frac{L}{2}\left\|\frac{\alpha_t}{\sqrt{v_t}}m_{t+1}\right\|_2^2 - \frac{\alpha_t}{2\sqrt{v_t}}\|\nabla f(\theta_t)\|_2^2 \\
&- \frac{\alpha_t}{2\sqrt{v_t}}\|m_{t+1}\|_2^2 \\
\leq& f(\theta_t) + \frac{\alpha_t b_{u,t}}{2}\Delta_t + \frac{L\alpha_t^2 b_{u,t}^2 - \alpha_t b_{l,t}}{2}\|m_{t+1}\|_2^2 - \frac{\alpha_t b_{l,t}}{2}\|\nabla f(\theta_t)\|_2^2.
\end{aligned} \tag{23}$$

Since we know that $T_0 \lesssim \frac{1}{\alpha_T}$, then we know the overall loss of the first T_0 terms would be $\mathbb{E}(\sum_{t=1}^{T_0}\|\nabla f(\theta)\|_2^2) \lesssim 1/\alpha_T$, and hence

$$\mathbb{E}\left(\frac{1}{T+1}\sum_{t=1}^{T_0}\|\nabla f(\theta)\|_2^2\right) \lesssim \frac{1}{\alpha_T(T+1)}. \tag{24}$$

For the other case when $t > T_0$, without loss of generality we can assume that $T_0 = 0$ for the above argument. We denote $A = \sqrt{\frac{b_{l,T}}{2L^2 b_{u,1}^3}}$ and $\theta^* = \arg\min_\theta f(\theta)$. From equation 23, we have

$$
\begin{aligned}
&\mathbb{E}\left(\sum_{t=0}^{T}\frac{\alpha_t b_{l,t}}{2}\left\|\nabla f(\theta_t)\right\|_2^2\right)\\
&\leq \mathbb{E}\left[\sum_{t=0}^{T}(f(\theta_t)-f(\theta_{t+1}))+\sum_{t=0}^{T}\frac{\alpha_t b_{u,t}}{2}\Delta_t+\sum_{t=0}^{T}\frac{(L\alpha_t^2 b_{u,t}^2-\alpha b_{l,t})}{2}\left\|m_{t+1}\right\|_2^2\right]\\
&\leq f(\theta_0)-f(\theta^*)+\mathbb{E}\left(\sum_{t=0}^{T}\frac{(L\alpha_t^2 b_{u,t}^2-\alpha b_{l,t})}{2}\left\|m_{t+1}\right\|_2^2\right)+\mathbb{E}\left(\sum_{t=0}^{T}\frac{\alpha_t b_{u,t}}{2(1-\beta_{1,t+1})}(1-\beta_{1,t+1})\Delta_t\right)\\
&\overset{\text{(i)}}{\leq} f(\theta_0)-f(\theta^*)+\mathbb{E}\left(\sum_{t=0}^{T}\frac{(L\alpha_t^2 b_{u,t}^2-\alpha b_{l,t})}{2}\left\|m_{t+1}\right\|_2^2\right)+\frac{Ab_{u,1}}{2}\mathbb{E}\left(\sum_{t=0}^{T}(1-\beta_{1,t+1})\Delta_t\right)\\
&\overset{\text{(ii)}}{\leq} f(\theta_0)-f(\theta^*)+\mathbb{E}\left(\sum_{t=0}^{T}\frac{(L\alpha_t^2 b_{u,t}^2-\alpha b_{l,t})}{2}\left\|m_{t+1}\right\|_2^2\right)\\
&\quad+\frac{Ab_{u,1}}{2}\mathbb{E}\left[\Delta_0+\sum_{t=0}^{T}2(1-\beta_{1,t+1})^2\sigma^2+\sum_{t=0}^{T}\frac{L^2\alpha_t^2 b_{u,t+1}^2}{1-\beta_{1,t+1}}\left\|m_{t+1}\right\|_2^2\right]\\
&\overset{\text{(iii)}}{\leq} f(\theta_0)-f(\theta^*)+\frac{Ab_{u,1}}{2}(\sigma^2+\|\nabla f(\theta_0)\|_2^2)+Ab_{u,1}\sigma^2\sum_{t=0}^{T}(1-\beta_{1,t+1})^2\\
&\quad+\mathbb{E}\left[\sum_{t=0}^{T}\left(\frac{AL^2 b_{u,1}\alpha_t^2 b_{u,t+1}^2}{2(1-\beta_{1,t+1})}+\frac{L\alpha_t^2 b_{u,t}^2-\alpha_t b_{l,t}}{2}\right)\left\|m_{t+1}\right\|_2^2\right]\\
&\overset{\text{(iv)}}{\leq} f(\theta_0)-f(\theta^*)+\frac{Ab_{u,1}}{2}(\sigma^2+\|\nabla f(\theta_0)\|_2^2)+Ab_{u,1}\sigma^2\sum_{t=0}^{T}(1-\beta_{1,t+1})^2,
\end{aligned}
$$

where (i) comes from the fact that $\alpha_t \leq (1-\beta_{1,t+1})A$ based on the conditions in Theorem 1; (ii) could be obtained after we apply equation 22 to the summation; (iii) is due to the fact that

$$
\begin{aligned}
\mathbb{E}(\Delta_0) &= \mathbb{E}(\|(1-\beta_{1,1})(g_1-\nabla f(\theta_0))-\beta_{1,1}\nabla f(\theta_0))\|_2^2)\\
&= (1-\beta_{1,1})^2\mathbb{E}(\|g_1-\nabla f(\theta_0)\|_2^2)+\beta_{1,1}^2\mathbb{E}\|\nabla f(\theta_0)\|_2^2 \leq \sigma^2+\|\nabla f(\theta_0)\|_2^2.
\end{aligned}
$$

And we can deduce (iv) by using the assumptions in Theorem 1

$$
\frac{AL^2 b_{u,1}\alpha_t^2 b_{u,t+1}^2}{2(1-\beta_{1,t+1})} \leq \frac{A^2L^2 b_{u,1}b_{u,t+1}^2}{2} \leq \frac{b_{l,t}}{4}, \quad \frac{L\alpha_t b_{u,t}^2}{2} \leq \frac{b_{l,t}}{4}.
$$

Therefore, we have

$$
\begin{aligned}
&\mathbb{E}\left(\sum_{t=0}^{T}\frac{\alpha_T b_{l,T}}{2}\left\|\nabla f(\theta_t)\right\|_2^2\right)\\
&\leq \mathbb{E}\left(\sum_{t=0}^{T}\frac{\alpha_t b_{l,t}}{2}\left\|\nabla f(\theta_t)\right\|_2^2\right)\\
&\leq f(\theta_0)-f(\theta^*)+\frac{Ab_{u,1}}{2}(\sigma^2+\|\nabla f(\theta_0)\|_2^2)+Ab_{u,1}\sigma^2\sum_{t=0}^{T}(1-\beta_{1,t+1})^2.
\end{aligned}
$$

As a consequence, we can deduce that

$$\begin{aligned}
&\mathbb{E}\left(\sum_{t=0}^{T}\frac{1}{T+1}\left\|\nabla f(\theta_t)\right\|_2^2\right)\\
&\leq \frac{\frac{2(f(\theta_0)-f(\theta^*))}{b_{l,T}}+\sqrt{\frac{1}{2L^2b_{u,1}b_{l,T}}}(\sigma^2+\|\nabla f(\theta_0)\|_2^2)}{\alpha_T(T+1)}+\sqrt{\frac{2}{Lb_{u,1}b_{l,T}}}\frac{\sigma^2\sum_{t=0}^{T}(1-\beta_{1,t+1})^2}{\alpha_T(T+1)}\\
&:= \frac{1}{\alpha_T(T+1)}(Q_1+Q_2\eta(T)),
\end{aligned} \tag{25}$$

where

$$Q_1=\frac{2(f(\theta_0)-f(\theta^*))}{b_{l,T}}+\sqrt{\frac{1}{2L^2b_{u,1}b_{l,T}}}(\sigma^2+\|\nabla f(\theta_0)\|_2^2),\quad Q_2=\sqrt{\frac{2}{Lb_{u,1}b_{l,T}}}\sigma^2.$$

□

B.3.3 Proof of Corollary 1

Without loss of generality we choose $1-\beta_{1,t}=\beta/\sqrt{t}$ and $\alpha_t=\alpha/\sqrt{t},\forall t\in[T]$ for some constants α,β with all conditions in Theorem 1 hold, we have

$$T\alpha_T=\alpha\sqrt{T},\quad \eta(T)=\sum_{t=1}^{T}(1-\beta_{1,t})^2=\beta^2\sum_{t=1}^{T}\frac{1}{t}\leq\beta^2(1+\log(T)).$$

After combining this with equation 25 and making some rearrangement, we have:

$$\begin{aligned}
&\mathbb{E}\left(\sum_{t=0}^{T}\frac{1}{T+1}\left\|\nabla f(\theta_t)\right\|_2^2\right)\\
&\leq \frac{\frac{2(f(\theta_0)-f(\theta^*))}{b_{l,T}}+\sqrt{\frac{1}{2L^2b_{u,1}b_{l,T}}}(\sigma^2+\|\nabla f(\theta_0)\|_2^2)}{\alpha\sqrt{T}}+\sqrt{\frac{2}{Lb_{u,1}b_{l,T}}}\frac{\sigma^2\beta^2(1+\log(T))}{\alpha\sqrt{T}}\\
&:= \frac{1}{\alpha_T(T+1)}(Q_1^*+Q_2^*\eta(T)),
\end{aligned}$$

where

$$\begin{aligned}
Q_1^*&=\frac{2(f(\theta_0)-f(\theta^*))}{b_{l,T}\alpha}+\sqrt{\frac{1}{2L^2b_{u,1}b_{l,T}}}\frac{(\sigma^2+\|\nabla f(\theta_0)\|_2^2)}{\alpha}+\sqrt{\frac{2}{Lb_{u,1}b_{l,T}}}\frac{\sigma^2\beta^2}{\alpha},\\
Q_2^*&=\sqrt{\frac{2}{Lb_{u,1}b_{l,T}}}\frac{\sigma^2\beta^2}{\alpha}.
\end{aligned}$$

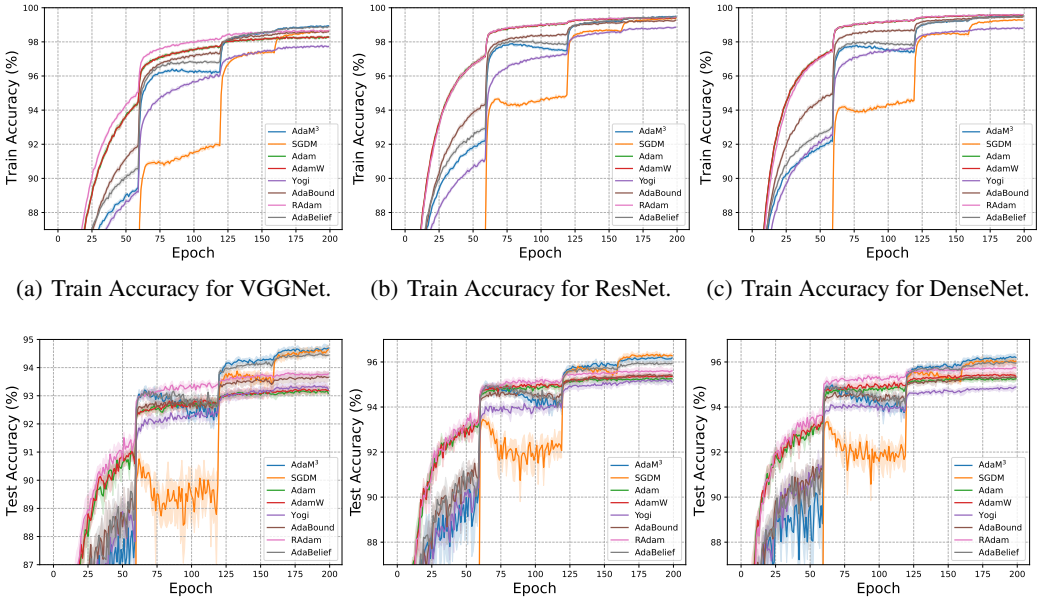
□

C Additional Experimental Details

C.1 Hyperparameter tuning rule

For hyperparameter tuning, we perform extensive and careful grid search to choose the best hyperparameters for all the baseline algorithms.

CNN for Image Classification For SGDM, we set the momentum as 0.9 which is the default choice (He et al., 2016; Huang et al., 2017) and search the learning rate between 0.1 and 10^{-5} in the log-grid. For all the adaptive gradient methods, we fix $\beta_1=0.9$ and $\beta_2=0.999$ and search the learning rate between 0.1 and $1e^{-5}$ in the log-grid, ϵ between $1e^{-5}$ and $1e^{-16}$ in the log-grid. For all optimizers we grid search weight decay parameter value in $\{1e^{-4},5e^{-4},1e^{-3},5e^{-3},1e^{-2},5e^{-2}\}$. For ImageNet, since we use cosine learning rate schedule, for SGDM we grid search the final learning rate in $\{1e^{-3},5e^{-4},1e^{-4},5e^{-5},1e^{-5}\}$ and for the adaptive gradient methods we search the final learning rate in $\{1e^{-5},5e^{-6},1e^{-6},5e^{-7},1e^{-7}\}$.



(a) Train Accuracy for VGGNet. (b) Train Accuracy for ResNet. (c) Train Accuracy for DenseNet.

(d) Test Accuracy for VGGNet. (e) Test Accuracy for ResNet. (f) Test Accuracy for DenseNet.

Figure 5: Train and test accuracy of different optimizers on CIFAR-10.

Table 7: Well tuned hyperparameter configuration of the adaptive gradient methods for CNNs on CIFAR-10.

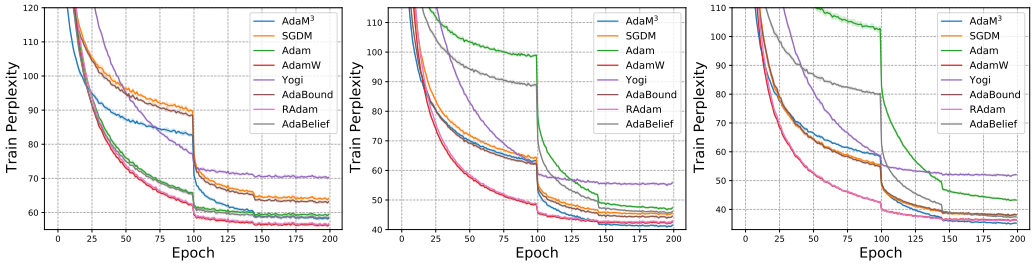
Algorithm	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief	AdaM^3
Stepsize α	0.001	0.001	0.001	0.001	0.001	0.001	0.001
β_1	0.9	0.9	0.9	0.9	0.9	0.9	0.9
β_2	0.999	0.999	0.999	0.999	0.999	0.999	0.999
Weight decay	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
ϵ	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-8}

LSTM for Language Modeling For SGDM, we grid search the learning rate in $\{100, 50, 30, 10, 10.1\}$ and momentum parameter between 0.5 and 0.9 with stepsize 0.1 . For all the adaptive gradient methods, we fix $\beta_2 = 0.999$ and search β_1 between 0.5 and 0.9 with stepsize 0.1, the learning rate between 0.1 and $1e^{-5}$ in the log-grid, ϵ between $1e^{-5}$ and $1e^{-16}$ in the log-grid. For all the optimizers, we fix the weight decay parameter value as $1.2e^{-4}$ following Zhuang et al. (2020).

Transformer for Neural Machine Translation For SGDM, we search learning rate between 0.1 and 10^{-5} in the log-grid and momentum parameter between 0.5 and 0.9 with stepsize 0.1. For adaptive gradient methods, we fix $\beta_1 = 0.9$, grid search β_2 in $\{0.98, 0.99, 0.999\}$, learning rate in $\{1e^{-4}, 5e^{-4}, 1e^{-3}, 1.5e^{-3}, 2e^{-3}, 3e^{-3}\}$, and ϵ between $1e^{-5}$ and $1e^{-16}$ in the log-grid. For all the optimizers we grid search weight decay parameter in $\{1e^{-4}, 5e^{-4}, 1e^{-3}, 5e^{-3}, 1e^{-2}, 5e^{-2}\}$.

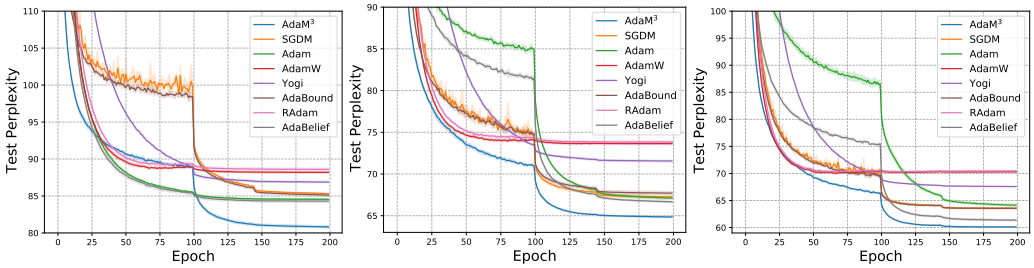
Generative Adversarial Network For SGDM we search the momentum parameter between 0.5 and 0.9 with stepsize 0.1. For all the adaptive gradient optimziers we set $\beta_1 = 0.5$, search β_2 and ϵ using the same schedule as previsou subsection.

All the experiments reported are trained on NVIDIA Tesla V100 GPUs. We provide some additional information concerning the empirical experiments for completeness.



(a) 1-Layer LSTM. (b) 2-Layer LSTM. (c) 3-Layer LSTM.

Figure 6: Train perplexity curve on Penn Treebank (Marcus et al., 1993) dataset.



(a) 1-Layer LSTM. (b) 2-Layer LSTM. (c) 3-Layer LSTM.

Figure 7: Test perplexity curve on Penn Treebank (Marcus et al., 1993) dataset.

C.2 Image Classification

CIFAR datasets The values of the hyperparameters after careful tuning of the reported results of the adaptive gradient methods on CIFAR-10 in the main paper is summarized in Table 7. For SGDM, the optimal hyperparameter setting is: the learning rate is 0.1, the momentum parameter is 0.9, the weight decay parameter is 5×10^{-4}. For Adabound, the final learning rate is set as 0.1 (matching SGDM) and the value of the hyperparameter gamma is 10^{-3}.

ImageNet For SGDM, the tuned stepsize is 0.1, the tuned momentum parameter is 0.9 and the tuned weight decay is 1×10^{-4}. For Adam, the learning rate is 0.001, $\epsilon = 1e^{-8}$, and the weight decay parameter is $1e^{-4}$. For AdaM^3, the learning rate is 0.001, $\epsilon = 1e^{-16}$ and the weight decay parameter is $5e^{-2}$.

C.3 LSTM on Language Modeling

The training and testing perplexity curves are illustrated in Figure 6 and 7. We can clearly see that AdaM^3 is able to make the perplexity descent faster than SGDM and most other adaptive gradient methods during training and mean while generalize much better in testing phase. In experimental settings, the size of the word embeddings is 400 and the number of hidden units per layer is 1150. We employ dropout in training and the dropout rate for RNN layers is 0.25 and the dropout rate for input embedding layers is 0.4.

The optimal hyperparameters of adaptive gradient methods for 1-layer, 2-layer and 3-layer LSTM are listed in Tables 8, 9 and 10 respectively. For SGDM, the Well tuned stepsize is 30.0 and the momentum parameter is 0.9. For Adabound, the final learning rate is set as 30.0 (matching SGDM) and the value of the hyperparameter gamma is 10^{-3}.

Table 8: Well tuned hyperparameter configuration of adaptive gradient methods for 1-layer-LSTM on Penn Treebank dataset.

Algorithm	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief	AdaM^3
Stepsize α	0.001	0.001	0.01	0.01	0.001	0.001	0.001
β_1	0.9	0.9	0.9	0.9	0.9	0.9	0.9
β_2	0.999	0.999	0.999	0.999	0.999	0.999	0.999
Weight decay	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}
ϵ	10^{-12}	10^{-12}	10^{-8}	10^{-8}	10^{-12}	10^{-16}	10^{-16}

Table 9: Well tuned hyperparameter configuration of adaptive gradient methods for 2-layer-LSTM on Penn Treebank dataset.

Algorithm	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief	AdaM^3
Stepsize α	0.01	0.001	0.01	0.01	0.001	0.01	0.001
β_1	0.9	0.9	0.9	0.9	0.9	0.9	0.9
β_2	0.999	0.999	0.999	0.999	0.999	0.999	0.999
Weight decay	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}
ϵ	10^{-12}	10^{-12}	10^{-8}	10^{-8}	10^{-12}	10^{-12}	10^{-16}

C.4 Transformer on neural machine translation

For transformer on NMT task, the well tuned hyperparameter values are summarized in Table 11. The stepsize of SGDM is 0.1 and the momentum parameter of SGDM is 0.9. Initial learning rate is 10^{-7} and the minimum learning rate threshold is set as 10^{-9} in the warm-up process for all the optimizers.

C.5 Generative Adversarial Network

The optimal momentum parameters of SGD for all GANs are 0.9. For adaptive gradient methods, the well tuned hyperparameter values for BigGAN with consistency regularization are summarized in Table 12. We implement the GAN experiments adapting the code from public repository [4]. We sample two visualization results of generated samples of GAN training with AdaM^3 in Figure 8.

D Hypotheis Test Results

Doing pair-wise hypothesis tests between our optimizer and each other optimizer helps performance comparison. Note that all the experiments reported in our main paper are run 5 times independently using random seeds. Since the sample size is comparatively small, we adopt non-parametric method Exact paired Wilcoxon signed rank test together with Paired t-test to conduct hypothesis tests on CIFAR image classification experiments (using best epoch test accuracy), LSTM language modeling experiments and BigGAN image generation experiment. The null hypothesis is that the compared baseline method is no worse than our AdaM^3 in performance, while the alternative hypothesis is that our proposed AdaM^3 is better than the compared baseline gradient method. The results (p-values) are summarized in Tab. 13-19. We can conclude from the tables that most of the P-values are quite small, which demonstrates that our shown superiority of AdaM^3 is reliable and universal.

[4] https://github.com/POSTECH-CVLab/PyTorch-StudioGAN

Table 10: Well tuned hyperparameter configuration of adaptive gradient methods for 3-layer-LSTM on Penn Treebank dataset.

Algorithm	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief	AdaM^3
Stepsize α	0.01	0.001	0.01	0.01	0.001	0.01	0.001
β_1	0.9	0.9	0.9	0.9	0.9	0.9	0.9
β_2	0.999	0.999	0.999	0.999	0.999	0.999	0.999
Weight decay	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}	1.2×10^{-4}
ϵ	10^{-12}	10^{-12}	10^{-8}	10^{-8}	10^{-12}	10^{-12}	10^{-16}

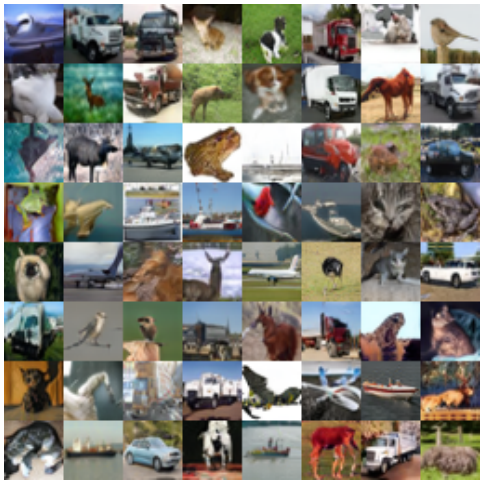
Table 11: Well tuned hyperparameter configuration of adaptive gradient methods for transformer on IWSTL'14 DE-EN dataset.

Algorithm	Adam	AdamW	AdaBelief	AdaM^3
Stepsize α	0.0015	0.0015	0.0015	0.0005
β_1	0.9	0.9	0.9	0.9
β_2	0.98	0.98	0.999	0.999
Weight decay	10^{-4}	10^{-4}	10^{-4}	10^{-4}
ϵ	10^{-8}	10^{-8}	10^{-16}	10^{-16}

Table 12: Well tuned hyperparameter configuration of adaptive gradient methods for BigGAN with consistency regularization.

Algorithm	Adam	Yogi	AdaBound	RAdam	AdaBelief	AdaM^3
β_1	0.5	0.5	0.5	0.5	0.5	0.5
β_2	0.999	0.999	0.999	0.999	0.999	0.999
Weight decay	0	0	0	0	0	0
ϵ	10^{-8}	10^{-8}	10^{-8}	10^{-8}	10^{-16}	10^{-16}

(a) DCGAN trained using random seed 0 (best FID score 43.52, iteration 26000).



(b) BigGAN trained using random seed 2 (best FID score 7.07, iteration 92000).

Figure 8: Generated figures trained on CIFAR-10 optimizing with AdaM^3.

Table 13: P-values (↓) calculated using pair-wise hypothesis tests between AdaM and the baseline optimizers on VGGNet-16 for CIFAR-10 classification.

Test type	SGDM	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief
Exact paired Wilcoxon signed rank test	0.313	0.031	0.031	0.031	0.031	0.031	0.031
Paried t-test	0.246	3.96e-6	8.34e-5	1.28e-4	1.66e-5	1.73e-5	0.012

Table 14: P-values (↓) calculated using pair-wise hypothesis tests between AdaM and the baseline optimizers on ResNet-34 for CIFAR-10 classification.

Test type	SGDM	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief
Exact paired Wilcoxon signed rank test	0.901	0.031	0.031	0.031	0.031	0.031	0.031
Paried t-test	0.902	4.35e-4	3.77e-4	1.17e-4	3.44e-4	2.59e-4	0.006

Table 15: P-values (↓) calculated using pair-wise hypothesis tests between AdaM and the baseline optimizers on DenseNet-121 for CIFAR-10 classification.

Test type	SGDM	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief
Exact paired Wilcoxon signed rank test	0.094	0.031	0.031	0.031	0.031	0.031	0.094
Paried t-test	0.112	2.17e-4	1.52e-3	4.99e-5	6e-4	0.003	0.04

Table 16: P-values (↓) calculated using pair-wise hypothesis tests between AdaM and the baseline optimizers on 1-layer LSTM for Penn Treebank dataset.

Test type	SGDM	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief
Exact paired Wilcoxon signed rank test	0.031	0.031	0.031	0.031	0.031	0.031	0.031
Paried t-test	4.54e-7	2.46e-6	2.21e-7	5.58e-7	6.07e-6	1.36e-5	1.36e-5

Table 17: P-values (↓) calculated using pair-wise hypothesis tests between AdaM and the baseline optimizers on 2-layer LSTM for Penn Treebank dataset.

Test type	SGDM	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief
Exact paired Wilcoxon signed rank test	0.031	0.031	0.031	0.031	0.031	0.031	0.031
Paried t-test	1.96e-5	2.50e-5	3.75e-9	1.88e-7	1.96e-5	2.16e-7	7.45e-6

Table 18: P-values (↓) calculated using pair-wise hypothesis tests between AdaM and the baseline optimizers on 3-layer LSTM for Penn Treebank dataset.

Test type	SGDM	Adam	AdamW	Yogi	AdaBound	RAdam	AdaBelief
Exact paired Wilcoxon signed rank test	0.031	0.031	0.031	0.031	0.031	0.031	0.031
Paried t-test	5.76e-6	3.75e-6	6.56e-8	2.26e-8	1.76e-6	1.16e-8	2.16e-4

Table 19: P-values (↓) calculated using pair-wise hypothesis tests between AdaM and the baseline optimizers on BigGAN experiments on CIFAR-10.

Test type	SGDM	Adam(W)	Yogi	AdaBound	RAdam	AdaBelief
Exact paired Wilcoxon signed rank test	0.031	0.031	0.031	0.031	0.031	0.094
Paried t-test	3.38e-7	1.01e-3	1.67e-4	3.68e-7	0.215	0.076