RMSPROP CAN CONVERGE WITH PROPER HYPER-PARAMETER

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the existence of divergence examples, RMSprop remains one of the most popular algorithms in machine learning. Towards closing the gap between theory and practice, we prove that RMSprop can converge with proper choice of hyper-parameters under certain conditions. More specifically, we prove that when the hyper-parameter β_2 is large enough, the random shuffling version of RMSprop converges to a bounded region in general, and converges to a stationary point in the interpolation regime. It is worth mentioning that our results do not depend on "bounded gradient" assumption, which is often the key assumption utilized by existing theoretical work for Adam. Removing this assumption allows us to establish a phase transition from divergence to non-divergence for RMSProp.

Finally, based on our theory, we conjecture that in practice there is a critical threshold th, such that RMSprop generates reasonably good results only if $\beta_2 \geq \mathsf{th}$. We provide empirical evidence for such a phase transition in our numerical experiments.

1 INTRODUCTION

RMSprop (Tielman & Hinton, 2012) remains one of the most popular algorithms for machine learning applications. As a non-momentum version of a more general algorithm Adam, RMSprop's good empirical performance has been well acknowledged by practitioners in generative adversarial networks (GANs) (Seward et al., 2020; Yazıcı et al., 2019; Karnewar & Wang, 2020; Jolicoeur-Martineau, 2019), reinforcement learning (Mnih et al., 2016), etc. In spite of its prevalence, however, Reddi et al. (2018) discovered that RMSprop (as well as the more general version Adam) can diverge even for simple convex functions. To fix the algorithm, the authors propose a new variant called AMSGrad, which is guaranteed to converge under certain conditions.

Since then, it has been an active area of research to design provably convergent variants of RMSprop. These variants include AdaFom (Chen et al., 2019), Adabound (Luo et al., 2019), Nostalgic Adam (Huang et al., 2019), Yogi (Zaheer et al., 2018), and many more. Despite the variants, the vanilla RMSprop indeed works well in practice, and after proper hyper-parameter tuning, the non-convergence issue has *not* been commonly observed. Why is there a large gap between theory and practice? Is this because the real-world problems are likely to be "nice", or is it because the theoretical analysis of RMSprop does not match how it is used in practice?

With the above questions in mind, we revisited the counter-example of (Reddi et al., 2018) and the convergence result of AMSGrad, and found an interesting discrepancy between them. One of the developed counter-examples is constructed for the following problem:

$$f_{\mathrm{t}}(x) = \begin{cases} Cx, & \text{for } t \bmod C = 1 \\ -x, & \text{otherwise} \end{cases} \tag{1}$$

defined for $x \in [-1, 1]$, with a hyper-parameter choice upper bounded by function of C: $\beta_2 \leq \min\{C^{-\frac{4}{C-2}}, 1 - \left(\frac{9}{2C}\right)^2\}$. For instance, when $C = 10$, then the algorithm diverges if $\beta_2 < 0.4$. But more careful experiments show there is a phase transition point for β_2 between 0.63 and 0.64: the algorithm will converge if β_2 is above this point. Reddi et al. (2018) mentioned that "this explains why large β_2 is advisable while using Adam algorithm", but it did not explain whether large β_2 leads to convergence in their example. We run simulation for problem (1) with different β_2 and found there

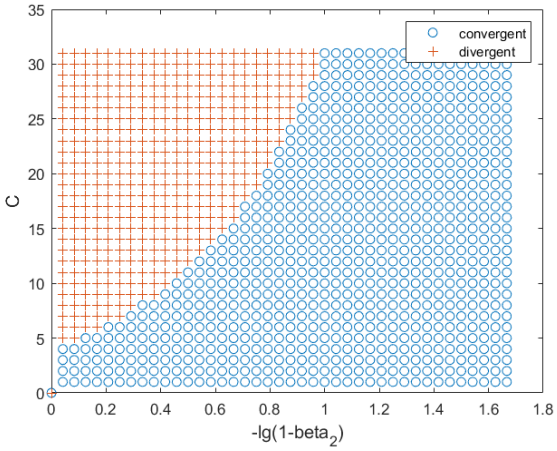

Figure 1: Phase diagram of the outcome of RMSprop on the counter example (1). Different marks represent different outcome: we label a data point as convergence if the distance between x and -1 is smaller than 0.01 on average after 1000 iterations and as divergence otherwise. For each choice of β_2, there exists a counter example, but for each counter example in which Adam diverges, there exists a larger β_2 that can make Adam converge. We fix β_1=0. Step size is set as $\eta_t = \frac{1}{\sqrt{t}}$

is always a threshold of β_2 above which RMSprop can converge, see Figure 1. Clearly, there is a curve of *phase transition* from divergence to convergence, and such a curve slopes upward, which means the transition point is closer to 1 if C becomes larger. Based on this observation, we make the following conjecture:

RMSprop converges if β_2 is large enough.

1.1 Main Contributions

To resolve the conjecture, we delve into RMSprop's convergence issues and obtain a series of theoretical and empirical results. We summarize our contributions below:

- We find RMSprop's convergence contingent on the choice of β_2. For general optimization problems, there are two types of hyper-parameters: *problem-dependent* hyper-parameters such as step size in GD, and *universal* constant such as momentum coefficient in heavy ball method [1]. Our result reveals that β_2 is closer to the first type.

- We prove that RMSprop converges to stationary point for realizable problems (interpolation regime) and to some bounded region for non-realizable problems. Combining with the divergence example of RMSProp, this indicates that there is a phase transition from divergence to convergence dependent on β_2; note that "convergence" is in a weak sense of converging to a bounded region for non-realizable case, and in a strong sense of converging to stationary points for realizable case.

- To our best knowledge, we are the first to prove the convergence of RMSprop and some of Adam *without* any form of assumption about the boundedness of the gradient norm. This is important for conveying our message of transition: with added assumptions on bounded gradients, then the gradients cannot diverge, while the counter-example shows that the gradient can diverge.

[1] Rigorously speaking, for the best convergence rate, the momentum coefficient should also be problem-dependent; but just for achieving convergence, it can be problem independent.

2 PRELIMINARIES

We start by considering the following general form of finite-sum problem:

$$\min_{x\in\mathbb{R}^d} f(x) = \sum_{j=0}^{n-1} f_j(x). \tag{2}$$

In neural network training, f_j usually represents loss contributed by the j-th sample batch. We present the randomly shuffled Adam algorithm as specified in Algorithm 1. We can simply set $\beta_1 = 0$ to recover RMSprop. We mainly focus on RMSProp, and will present a relatively weak result on Adam (a subset of Adam).

Algorithm 1 Randomly Shuffled Adam

Initialize $m_{1,-1} = \frac{1}{(1-\beta_1)}\nabla f(x_0)$ and $v_{1,-1} = \frac{1}{1-\beta_2}\max_j\{\nabla f_j(x_0)\circ\nabla f_j(x_0)\}$
for $k = 1 \to \infty$ **do**
  Sample $\{\tau_{k,0}, \tau_{k,1}, \cdots, \tau_{k,n-1}\}$ as a random permutation of $\{0, 1, 2, \cdots, n-1\}$
  for $i = 0 \to n-1$ **do**
    $m_{k,i} = \beta_1 m_{k,i-1} + (1-\beta_1)\nabla f_{\tau_{k,i}}$
    $v_{k,i} = \beta_2 v_{k,i-1} + (1-\beta_2)\nabla f_{\tau_{k,i}} \circ \nabla f_{\tau_{k,i}}$
    $x_{k,i+1} = x_{k,i} - \frac{\eta_{k*n}}{\sqrt{v_{k,i}+\epsilon}} \circ m_{l,k,i}$
    Break if certain exit condition is satisfied.
  end for
  $x_{k+1,0} = x_{k,n}$
  $v_{k+1,-1} = v_{k,n-1}$
  $m_{k+1,-1} = m_{k,n-1}$
end for
return x

In Algorithm 1, x is the iterate, m denotes the gradient estimate and v is used to inversely weight the gradient estimate. Specifically, we denote $x_{k,i}, m_{k,i}, v_{k,i} \in \mathbb{R}^d$ as the value of x, m, v at the k-th outer loop and i-th inner loop, respectively. We also denote ∇f_j as the gradient of f_j and let $\circ$ be the component-wise multiplication. Division of two vectors is component-wise too. Moreover, we denote η_t as the step-size and β_1, β_2 as the hyper-parameters in the algorithm. When $n = 1$, we obtain full batch Adam. We replaced the bias correction step in (Kingma & Ba, 2015) with a special initialization on $m_{1,-1}$ and $v_{1,-1}$. This initialization can also correct the bias, but have cleaner results. Since the effect of initialization or bias correction becomes more and more negligible as the training progresses, RMSprop with zero initialization or our initialization will have the same asymptotic behavior. We put our results for the original version of Adam in the appendix.

Let us discuss the the choice of parameters. We choose $\eta_t = \frac{\eta_1}{\sqrt{t}}$, and fix β_2 as constant that is independent of the iterations. This simple setting is standard in practice while contrary to the existing theoretical results (Chen et al., 2019), where they chose iteration-dependent hyper-parameters $\beta_{2,t}$. In our theoretical analysis, we will downplay the effect of the constant ϵ, which is usually added for numerical stability. The reason is that ϵ is typically chosen to be 10^{-6} or even 10^{-8}, so it is much smaller compared with $\sqrt{v_{k,i}}$ (the size of gradient norm). Reference De et al. (2019) remarked that ϵ is crucial in their approaches of proving convergence. In comparison, our results are independent of ϵ and thus able to show that RMSprop can converge even without ϵ.

2.1 RELATED WORK

As has been discussed above, one line of research focuses on mutants of RMSprop and Adam that can be theoretically proved to converge. These works usually modify the update rule on v_t: AdaGrad, AMSgrad (Reddi et al., 2018), AdaFom (Chen et al., 2019) explicitly make v_t non-decreasing; Nostalgic Adam (Huang et al., 2019) and algorithm analyzed in (Zou et al., 2019) used iteration-dependent β_{1t} and β_{2t} to let v_t weigh more on past gradients. Some authors added new motivations into RMSprop and Adam, like (Zhou et al., 2019) mitigated the bias in update direction by using a different estimate of v_t, (Dozat, 2016) combined Adam with Nesterov method, (Liu et al., 2019)

employed a warm-up technique to escape bad local minima. However these algorithms are usually more intuition based and less theoretically grounded.

Besides, there are still many papers on the theoretical side attempting to address the non-convergence issue. These works typically rely on additional (and often unrealistic) assumptions. For example, a number of recent works such as (Zaheer et al., 2018), and (De et al., 2019) analyzed Adam based on the bounded (stochastic) gradient assumption, and they require that ϵ added in denominator of the term $\frac{\eta_{k*n}}{\sqrt{v_{k,i}}}$ be large. Specifically, these convergence rates depend on ϵ and become infinitely large in the limit $\epsilon \to 0$. Therefore, to achieve a reasonable rate, ϵ has to be relatively large compared to $\sqrt{v_{k,i}}$. However, such a choice essentially transforms RMSprop back to SGD since the effective step size is controlled by ϵ, not $\sqrt{v_{k,i}}$. This is in contrary to the spirit of RMSprop, which is to use adaptive step size to accelerate convergence. Another example is (De et al., 2018), which analyzes deterministic and stochastic RMSprop algorithm. Again, they relied on two rather unrealistic assumptions: the bounded gradient assumption, and the sign of all noisy gradients are the same: $\text{sign}(\nabla f_p(x)) = \text{sign}(\nabla f_q(x))$ for all p, q. Additionally, reference (Chen et al., 2019) described a few quantities based on the iterates, and proved that if they grow in a certain speed as the iterates go, the algorithm converges. The drawback is that the condition cannot be checked *a priori*.

Very recently, we notice a concurrent work (Liu et al., 2020) improved theoretical analysis of SGDM and obtained satisfactory rates also without the bounded gradient assumption. However, they do not discuss RMSPprop.

3 THE *raison d'être* FOR β_2

From the experiment in Figure 1, it is clear that the parameter β_2 plays an important role in the convergence of RMSprop. Specifically, a sufficiently large β_2 is critical for RMSprop's convergence. Indeed, some recent works (Reddi et al., 2018) (Zhou et al., 2019) have also made analogous arguments, but they focused on understanding one part of the phenomenon, that is, small β_2 leads to divergence. Our goal in this work is to complete the other part of the story by showing that, sufficiently large β_2 guarantees convergence. This result will be provided in Sec. 4.

To understand the role of β_2, let us first discuss why RMSprop diverges. It is known that the stochastic noise due to mini-batch will distort the gradient direction, leading to possible divergence. Specifically, at a given iteration the scaling constant $1/\sqrt{v}$ in the update direction may cause larger gradient distortion than the standard SGD. The distortion can be so significant that the average updating direction falls outside the dual cone of the true gradient. To illustrate this, consider the extreme case that $\beta_2 = 0$ (i.e., signSGD). When applying signSGD to solve (1), in each epoch of C iterations, one iteration will move x left followed by $C - 1$ iterations that move x right. Since all step sizes are the same in one epoch, the accumulated effect of one epoch makes x move in the ascending direction, instead of the descending direction.

Then why does large β_2 help? Intuitively, a large β_2 can control the distortion on update directions. In the extreme case that $\beta_2 = 1$, RMSprop reduces to SGD, thus can converge. Our experiment in Figure 1 also confirms that, at least for the counter-example proposed in Reddi et al. (2018), there is an interval $\beta_2 \in [c, 1]$ such that RMSprop converges.

To make the above intuition concrete, we divide problems into 2 sub-classes: *realizable* problems and *non-realizable* problems to rigorously study the effect of β_2. The first class of problem, also known as the interpolation scheme, has the property that at a global optimal solution, all local objective is also zero. The choice of such a problem class is motivated by the following facts: 1) The example (1) provided in (Reddi et al., 2018) is indeed realizable; 2) this class of problem often appears in large and over-parameterized neural networks, for which the RMSprop/Adam algorithm works well (Choi et al., 2020). We show that RMSprop converges for this class of problem, as long as β_2 is large enough, and the step sizes are chosen appropriately. For *non-realizable* problems, we discovered an example for which RMSprop cannot converge to optimal solution for a wide range of $\beta_2 < 1$. Additionally, we perform several numerical experiments to understand the boundary of convergence behavior for realizable vs non-realizable, and large vs small β_2; see Table 1 for a generic pattern. These results combined together suggest that the realizable condition is (almost) sufficient for the convergence (to critical points) of RMSprop.

Setting	β_2 close to 1	β_2 close to 0
non-realizable	oscillate	diverge
realizable	converge	diverge

Table 1: Outcome of random shuffled RMSprop under different parameter setting. On unconstrained problems, if gradient norm decreases steadily for larger than 10^7 epochs on average, we identify the result as convergence. If gradient norm increases steadily, we identify the result as divergence. We identify a result as oscillating if gradient norm converges to a value larger than zero, i.e. if initial gradient norm is above this level, it decreases in later iterations and if below, increases.

As justifications for our intuition and numerical findings, we prove that a sufficiently large β_2 can help algorithm converge into bounded region whose size can be controlled by β_2. We also prove that for a specific realizable problem, there exists a data-dependent threshold of β_2 above which RMSprop can converge into stationary point. Note that this result does not conflict with Theorem 3 in Reddi et al. (2018) which claims that "for any constant β_1 and β_2 there exists a divergent example" since here we choose β_2 for a specific model, just like one chooses a step size $< 2/L$ for GD where L is a problem dependent parameter. Another important remark is that though β_2 could be close to 1, RMSprop still retains the ability to adapt v to gradient square norm as long as $\beta_2 < 1$, because new gradient signal are added for each iteration and the impact of previous signals decays exponentially. It is the adaptive ability that distinguishes RMSprop from SGD.

4 Convergence Results

We will explain the details of our results in this section, starting from full batch RMSprop/Adam, then moving to the random shuffled versions.

4.1 Full-Batch

Diminishing step size is necessary for RMSprop to converge to fixed point (Chen et al., 2019). We consider one popular setting of step size: $\eta_t = \frac{\eta_1}{\sqrt{t}}$. The following theorem shows that if we use all samples to evaluate gradient, diminishing step size is sufficient for convergence, regardless of the choice of β_2:

Theorem 4.1. *(convergence of full-batch RMSprop) For problem (2) with $n = 1$, assume that f is gradient Lipschitz continuous with constant L and lower bounded by f^*. Then, for full-batch RMSprop (Alg. 1 with $\beta_1 = 0$) with diminishing step size $\eta_t = \frac{\eta_1}{\sqrt{t}}$ and any $\beta_2 \in (0, 1)$, we have:*

$$\min_{t\in(1,T]} \|\nabla f_t\|_1 \leq \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$$

where $T > 0$ is the total iteration number.

Unlike in (De et al., 2019) that relies on a gradient norm upper bound to prove the convergence of full batch RMSprop, our result only requires lower-boundedness of the function value and the gradient Lipschitz continuity.

Our result suggests that, RMSprop is similar to the signSGD, an algorithm that only uses the sign of gradient to calculate its descent direction (Bernstein et al., 2018). This is a part of the motivation for authors who originally proposed RMSprop. In the full-batch setting, signSGD (which can be called sign GD) is also proved to converge without bounded gradient assumption.

Below, we also derive an analogous result for full-batch Adam with only one additional constraint $\beta_1 < \sqrt{\beta_2} < 1$, which is often satisfied in practice:

Theorem 4.2. *(convergence of full-batch Adam) For optimization problem (2) with $n = 1$, assume that f is gradient Lipschitz continuous with constant L and lower bounded by f^*. Then, for full-batch Adam with diminishing step size $\eta_t = \frac{\eta_1}{\sqrt{t}}$ and any $\beta_1 < \sqrt{\beta_2} < 1$, we have:*

$$\min_{t\in(1,T]} \|\nabla f_t\|_1 \leq \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$$

4.2 RANDOMLY SHUFFLED VERSIONS

As mentioned earlier, our simulation shows that RMSprop may not converge to critical points for non-realizable problems (an example is provided in the appendix). Nevertheless, we can still show random shuffling RMSprop converges to bounded region with large enough β_2:

Theorem 4.3. *(large-β_2 RMSprop always converge to bounded region) For optimization problem (2), assume that f is lower-bounded by f^* and f_j is gradient Lipschitz continuous with constant L for all j. Furthermore, assume that f_i's satisfy the following*

$$\sum_{j=0}^{m-1} \|\nabla f_j(x)\|_2^2 \leq D_1 \|\nabla f(x)\|_2^2 + D_0. \tag{3}$$

Then, for random-shuffle RMSprop with diminishing step size $\eta_t = \frac{\eta_1}{\sqrt{t}}$ and β_2 that satisfies,

$$T_2(\beta_2) \triangleq \sqrt{\frac{5d}{\beta_2^n}} dn^2 D_1 \left((1-\beta_2) \frac{\left(\frac{4n^2}{\beta_2^n} - 1\right)}{2} + \left(\frac{1}{\sqrt{\beta_2^n}} - 1 \right) \right) \leq \frac{\sqrt{2}-1}{2\sqrt{2}}, \tag{4}$$

we have

$$\min_{t\in[1,T]} \min\{\|\nabla f_{nt}\|_1, \|\nabla f_{nt}\|_2^2 \sqrt{\frac{D_1 d}{D_0}}\} \leq \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right) + \mathcal{O}\left(Q_{3,3}\sqrt{D_0}\right), \ \forall\, T \geq 4.$$

Here $Q_{3,3} > 0$ is a constant that goes to zero in the limit as $\beta_2 \to 1$.

Remark 1. This result and the result in Reddi et al. (2018) together distinguish large-β_2-RMSprop and small-β_2-RMSprop: the former converges to a bounded region, while the latter can diverge. Note that there is a gap between the lower bound of β_2 and the upper bound of β_2 in the counter-example. We do not try to provide tight bounds on the threshold of β_2 , as our main goal is to show a qualitative difference between large-β_2-RMSprop and small-β_2-RMSprop. We leave the closure of the gap to future work.

Remark 2. We point out three possible algorithm behavior: divergence to infinity (or divergence for short), convergence to a bounded region (or non-divergence for short) and convergence to critical points. We distinguish the three cases, making it easier to explain the qualitative difference of small-β_2 and large-β_2 regime. For non-realizable cases, the phase transition is from divergence to non-divergence. Therefore, it is important to discard the bounded-gradient assumption: this assumption eliminate the possibility of divergence of gradients a priori. To be clear, there are actually two sub-cases of non-divergence cases: proving a meaningless bound that the iterates can stay in a bounded but huge region, or proving a meaningful bound that the iterates stay in a bounded region dependent on some parameters. Indeed, the "convergence" of constant-stepsize SGD is in the sense of "converging to a region with size proportional to the noise variance". Our result of "converging to bounded region" is also meaningful as the size of the region goes to zero as the noise variance goes to 0 or D_0 goes to 0 (realizable case).

Note that "divergence" can be also interpreted as "not converging to critical points" which is the notion used in (Reddi et al., 2018), instead of "diverging to infinity". We use the latter concept of "diverging to infinity" for the term "divergence", because "not converging to critical points" can include the good case of converging to a small region around critical points (like constant-stepsize SGD). In the example of Reddi et al. (2018), a constrained problem is considered (bound constraint [-1,1]), thus divergence to infinity cannot happen. We add an example that the iterates and the gradients can diverge to infintity for small β_2; see Appendix A.2.

The realizable assumption becomes popular recently partially because researchers realize that the neural-nets used in practice can interpolate all data points (e.g., due to over-parameterization). Realizable problem can be regarded as a special case of theorem 4.3, with $D_0 = 0$. The assumption $\sum_{j=0}^{m-1} \|\nabla f_j\|_2^2 \leq D_1 \|\nabla f\|_2^2$ is called strong growth condition. Proposed in (Vaswani et al., 2019), it is a natural extension of realizability. It requires the noisy gradient norm be proportional to the true gradient norm. When $\|\nabla f\| = 0$, strong growth condition implies $\|\nabla f_j\| = 0$ for all j, and thus the problem is realizable. Many problems satisfy strong growth condition. For instance, a generic

unconstrained quadratic optimization problem is

$$\min_{\mathbf{x}} \|A\mathbf{x}\|^2 = \sum_{j=1}^{n} (a_j \mathbf{x})^2$$

where A is an n by n matrix and a_j is the j-th row vector of A. It is easy to see that the problem satisfies the strong growth condition with $D_1 \leq \lambda_{\max}\left(\sum_{i=1}^{n} a_i^T a_i a_i^T a_i\right) / \lambda_{\min}\left(A^T A\right)$ (Raj & Bach, 2020). Theorem 4.3 shows if the optimization problem satisfies strong growth condition, RMSprop can converge to a fixed point. See the following corollary:

Corollary 4.1. *For optimization problem (2), assume that f is lower-bounded by f^* and f_i is gradient Lipschitz continuous with constant L for all i. Furthermore, assume that f_j satisfies the strong growth condition $\sum_{j=0}^{m-1} \|\nabla f_j(x)\|_2^2 \leq D_1 \|\nabla f(x)\|_2^2$ for all x. Then, for random shuffle RMSprop with step size $\eta_t = \frac{\eta_1}{\sqrt{t}}$ and sufficiently large (but constant) β_2, we have:*

$$\min_{t \in [1,T]} \|\nabla f_{nt}\|_1 \leq \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right), \ \forall\, T \geq 4.$$

With the above corollary, the numerical result in Figure 1 should not be surprising: problem (1) satisfies the strong growth condition, thus there is always a range of β_2 inside which RMSprop converges. We just need to tune β_2 larger.

Similar results also exist for Adam with small β_1. The following theorem is the result about the convergence of Adam. We conjecture that the same convergence can be proved for any β_1, but we are not able to prove that for now (which is why we focus on RMSProp in this work) and leave it to future work.

Theorem 4.4. *For optimization problem (2), assume that f is lower-bounded by f^* and f_j is gradient Lipschitz continuous with constant L for all j. Furthermore, assume that f_j satisfies* (3) *for all x. Then, for randomly shuffled Adam with diminishing step size $\eta_t = \frac{\eta_1}{\sqrt{t}}$ and β_1, β_2 satisfying*

$$T_1(\beta_1, \beta_2) + T_2(\beta_2) < 1 - \frac{1}{\sqrt{2}} \tag{5}$$

we have $\min_{t \in [1,T]} \|\nabla f_{nt}\|_1 \leq \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right) + \mathcal{O}\left(Q_{3,5}\sqrt{D_0}\right)\ \forall\, T \geq 4$, where $Q_{3,5}$ is a constant that approaches 0 *in the limit $T_1 + T_2 \to 1$, T_2 is defined in* (4)*, and T_1 is defined as:*

$$T_1(\beta_1, \beta_2) = \sqrt{\frac{5d}{\beta_2^n}} d n^2 D_1 \frac{\beta_1}{\beta_2^n}\left(\frac{1-\beta_1}{1-\beta_1^n} + 1\right) \tag{6}$$

This result show that controlling β_2 and β_1 together is sufficient for curbing gradient signal distortion and for convergence.

5 EXPERIMENTS

We conduct some experiments on GAN and image classification using data sets including MNIST and CIFAR-10 to verify our theoretical findings. GAN experiment details are in the appendix. For the MNIST experiment, we visualize the optimization trajectory for large $\beta_2 = 0.8$ and $\beta_2 = 0.99$ in Figure 2, and observe different behavior.

In the CIFAR experiments, we used Resnet18 as our model. We choose $\beta_2 = 0.8, 0.9, 0.95, 0.99$ respectively. With different batch sizes $8, 16, 32$, we run each algorithm for 100 epochs without explicit regularization. As the results shown in Table 2, when the batch size is fixed, both training and test accuracy increase as β_2 becomes larger. It is also clear that the transition point of β_2 where training and test accuracy increase significantly becomes lower as the batch size becomes larger. We apply our theoretical result to provide an intuition: when the batch size is larger, the optimization problem can be roughly regarded as an equivalent problem with fewer sample batches n. Since in Theorem 4.3, T_2 decreases as n decreases, we can choose smaller β_2 while still guarantee that (8) holds.

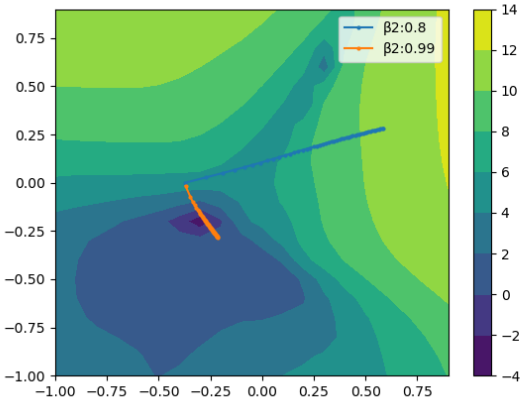

Figure 2: Trajectories of RMSprop with large or small β_2 and loss surface. Data points represent network weights obtained by training a convolutional neural network on MNIST. Two trajectories start from the same initialization and are trained on the same dataset, but have very different behavior. We calculate the hyper-plane spanned by the starting point of two trajectories and their respective ending points, then project two trajectories in this plane. The loss surface represents the logarithm of the average cross entropy loss evaluated on the training set. One can easily observe that large β_2 trajectory converge into minima while small β_2 trajectory diverge. The difference between these two trajectories suggests a phase transition of β_2 in $(0.8, 0.99)$

Table 2: Performance of Adam with different β_2 with Resnet-18 on CIFAR-10 (100 epochs)

batch size	measure	β_2=0.8	β_2=0.9	β_2=0.95	β_2=0.99	SGD
8	train acc.	10.00±0.00	10.00±0.00	44.53±32.09	99.74±0.06	100.00±0.00
	test acc.	10.00±0.00	10.00±0.00	42.02±29.59	70.23±0.26	70.37±0.45
16	train acc.	28.70±32.39	67.27±8.98	96.38±1.35	99.75±0.05	99.98±0.02
	test acc.	27.64±30.55	62.71±7.71	70.11±0.90	70.43±0.15	69.45±0.38
32	train acc.	66.93±3.07	96.72±1.36	99.17±0.42	99.80±0.14	81.50±1.57
	test acc.	62.99±2.13	70.05±1.40	71.92±0.50	71.34±0.60	68.92±1.12

One may observe from Table 2 that SGD seems to have more stable convergence when the batch size is small. Nevertheless, the convergence speed of SGD is much slower than Adam. This is demonstrated in Table 3 when we compare the average training and test accuracy at the 10-th epoch.

Table 3: Training and test accuracy at the 10-th epoch

batch size	measure	β_2=0.99	SGD
16	train acc.	95.41±0.81	65.89±1.28
	test acc.	70.02±0.17	62.62±1.26
32	train acc.	97.92±0.23	57.87±0.70
	test acc.	70.44±0.19	56.18±0.86

All codes generating experimental results are available on the anonymous repository `https://anonymous.4open.science/r/f6cfb2f1-093c-4808-86b0-d2af4cd10afb/`

6 Conclusion

In this work, we study the convergence behavior of RMSprop by taking a closer look at the hyper-parameters. Specifically, for realizable problems, we provide a data-dependent threshold of β_2 above which we prove the convergence of randomly shuffled RMSprop and small β_1 Adam without bounded gradient assumption. We also show that RMSprop converge into a bounded region under non-realizable settings. These findings reveal that there is a critical threshold of β_2 regarding the convergence behavior of RMSprop, and the phase transition is supported by the numerical experiments. Our results provide basic guidelines for tuning hyper-parameters in practice.

REFERENCES

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Anima Anandkumar. Signsgd: Compressed optimisation for non-convex problems. *Arxiv*, 2018.

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

Dami Choi, Christopher J. Shallue, Zachary Nado, Jaehoon Lee, Chris J. Maddison, and George E. Dahl. On empirical comparisons of optimizers for deep learning. *Arxiv*, 2020.

Soham De, Anirbit Mukherjee, and Enayat Ullah3. Convergence guarantees for rmsprop and adam in non-convex optimization and an empirical comparison to nesterov acceleration. *Arxiv*, 2018.

Soham De, Anirbit Mukherjee, and Enayat Ullah. Understanding rmsprop and adam: Theoretical and empirical studies. *Arxiv*, 2019.

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

Haiwen Huang, Chang Wang, and Bin Dong. Nostalgic adam: Weighting more of the past gradients when designing the adaptive learning rate. *IJCAI*, 2019.

Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard gan. *ICLR*, 2019.

Animesh Karnewar and Oliver Wang. Msg-gan: Multi-scale gradients for generative adversarial networks. *CVPR*, 2020.

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

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. *Arxiv*, 2019.

Yanli Liu, Yuan Gao, and Wotao Yin. An improved analysis of stochastic gradient descent with momentum. *NIPS*, 2020.

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

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *ICML*, 2016.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *ICLR*, 2016.

Anant Raj and Francis Bach. Explicit regularization of stochastic gradient methods through duality. *Arxiv*, 2020.

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

Calvin Seward, Thomas Unterthiner, Urs Bergmann, Nikolay Jetchev, and Sepp Hochreiter. First order generative adversarial networks. *Arxiv*, 2020.

T. Tielman and G. Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. *Coursera*, 2012.

Sharan Vaswani, Francis Bach, and Mark Schmidt. Fast and faster convergence of sgd for over-parameterized models (and an accelerated perceptron). *NIPS*, 2019.

Yasin Yazıcı, Chuan-Sheng Foo, Stefan Winkler, Kim-Hui Yap, Georgios Piliouras, and Vijay Chandrasekhar. The unusual effectiveness of averaging in gan training. *ICLR*, 2019.

Manzil Zaheer, Sashank J.Reddi, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. Adaptive methods for nonconvex optimization. *NIPS*, 2018.

Zhiming Zhou, Qingru Zhang, Guansong Lu, Hongwei Wang, Weinan Zhang, and Yong Yu. Adashift: decorrelation and convergence of adaptive learning methods. *ICLR*, 2019.

Fangyu Zou, Li Shen, Zequn Jie, Weizhong Zhang, and Wei Liu. A sufficient condition for convergences of adam and rmsprop. *CVPR*, 2019.