Divergence Results and Convergence of a Variance Reduced Version of ADAM

Anonymous Author
Anonymous Institution

Abstract

Stochastic optimization algorithms using exponential moving averages of the past gradients, such as ADAM, RMSProp and AdaGrad, have been having great successes in many applications, especially in training deep neural networks. ADAM in particular stands out as efficient and robust. Despite of its outstanding performance, ADAM has been proved to be divergent for some specific problems. We revisit the divergent question and provide divergent examples under stronger conditions such as in expectation or high probability. Under a variance reduction assumption, we show that an ADAM-type algorithm converges, which means that it is the variance of gradients that causes the divergence of original ADAM. To this end, we propose a variance reduced version of ADAM and provide a convergent analysis of the algorithm. Numerical experiments show that the proposed algorithm has as good performance as ADAM. Our work suggests a new direction for fixing the convergence issues.

1 Introduction

Stochastic optimization based on mini-batch is a common training procedure in machine learning. Suppose we have finitely many differentiable objectives $\{f_n(w)\}_{n=1}^N$ defined on $\mathbb{R}^d$ with N being the size of the training set. In each iteration, a random index set $\mathcal{B}_t$ is selected from $\{1,\ldots,N\}$ and the update is made based on the mini-batch loss $F^{\mathcal{B}_t}(w) = \frac{1}{b}\sum_{n\in\mathcal{B}_t} f_n(w)$, where $b = |\mathcal{B}_t|$ is the batch size. The goal is to minimize the empirical risk $\min_{w\in\mathbb{R}^d} F(w) := \frac{1}{N}\sum_{n=1}^N f_n(w)$.

First order methods, which make updates based on the information of the gradient of mini-batch loss functions, prevail in practice, [Goodfellow et al., 2016]. A simple method is stochastic gradient descent (SGD), where the model parameters are updated at the negative direction of the mini-batch loss gradient in each iteration. Although SGD is straightforward and is proved to be convergent, the steps of SGD near the minima are very noisy and take longer to converge. Several adaptive variants of SGD, such as AdaGrad [Duchi et al., 2011], RMSProp [Hinton et al., 2012] and ADAM [Kingma and Ba, 2015], are proved to converge faster than SGD in practice. These methods take the historical gradients into account. Specifically, instead of using a predefined learning rate schema, they adjust the step size automatically based on the information from the past mini-batch losses. AdaGrad is the earliest algorithm in the adaptive method family and performs better than SGD when gradients are sparse. Although AdaGrad has great theoretical properties for convex loss, it does not work well practically in training. RMSProp replaces the sum of square scaling in AdaGrad with exponential moving average and fixes the rapid decay of the learning rate in AdaGrad. ADAM-type algorithms combine the exponential moving average of both first and second order moments. The original ADAM enjoys the advantages of AdaGrad in sparse problems and RMSProp in non-stationary problems and became one of the most popular optimization methods in practice.

Yet, ADAM may fail to solve some problems. Reddi et al. [Reddi et al., 2018] found a flaw in the proof of convergence in [Kingma and Ba, 2015] and proposed a divergent example for online ADAM. Based on the divergent example, they pointed out that when some large, informative but rare gradients occur, the exponential moving average would make them decay quickly and hence would lead to the failure of convergence. To this end, Reddi et al. proposed two variants of ADAM to fix this problem. The first proposal, known as AMSGrad, suggests taking the historical maximum of the ADAM state v_t in order to obtain 'long-term memories' and prevent the large and informative gradients from being forgotten. Although this helps keeping the information of large gradients, it hurts the adaptability of ADAM. If the algorithm is exposed to a large gradient at early iterations, the v_t parameter will stay constant, hence the algorithm will not automatically adapt the step size, and it will degenerate to a momentum method. Another intuitive criticism is that keeping v_t increasing is against what one expects, since if the algorithm converges, the norm of gradients should

Preliminary work. Under review by AISTATS 2023. Do not distribute.

decrease and $v_{t+1} - v_t = (1-\beta_2)(g_t^2 - v_t)$ is more likely to be negative, where g_t is the stochastic gradient in step t and β_2 is a hyper parameter.

Several other proposals tried to fix the divergent problem of ADAM. The second variant proposed in [Reddi et al., 2018], called ADAMNC, requires the second order moment hyper-parameter β_2 to increase and to satisfy several conditions. However the conditions are hard to check. Although they claim that $\beta_{2,t} = 1 - 1/t$ satisfies the conditions, this case is actually AdaGrad, which is already well-known for its convergence. Zhou et al. [Zhou et al., 2019] analyzed the divergent example in [Reddi et al., 2018], and pointed out that the correlation of v_t and g_t causes divergence of ADAM, and proposed a decorrelated variant of ADAM. The theoretical analysis in [Zhou et al., 2019] is based on complex assumptions and they do not provide a convergence analysis of their algorithm. Several other works, such as [Guo et al., 2021, Shi et al., 2020, Wang et al., 2019, Zou et al., 2019] suggested properly tuning the hyper-parameters of ADAM-type algorithms had helped with convergence in practice.

It is empirically well-known that larger batch size reduces the variance of the loss of a stochastic optimization algorithm. [Qian and Klabjan, 2020] gave a theoretical proof that the variance of the stochastic gradient is proportional to $1/b$. Although several works connected the convergence of ADAM with the mini-batch size, the direct connection between convergence and variance is wanted. For the full-batch case (i.e., where there is no variance), [De et al., 2018] showed that ADAM converges under some specific scheduling of learning rates. [Shi et al., 2020] showed the convergence of full gradient ADAM and RMSProp with the learning rate schedule $\alpha_t = \alpha/\sqrt{t}$ and constants β_1 and β_2 satisifying $\beta_1 < \sqrt{\beta_2}$. For the stochastic setting with a fixed batch size, Zaheer et al. [Zaheer et al., 2018] proved that the expected norm of the gradient can be bounded into a neighborhood of 0, whose size is proportional to $1/b$. They suggested to increase the batch size with the number of iterations in order to establish convergence. One question is that whether there exists a threshold of batch size $b^* < N$, such that any batch size larger than b^* guarantees convergence. We show that even when $b = N - 1$, there still exist divergent examples of ADAM. This means that although large batch size helps tighten the optimality gap, the convergence issue is not solved as long as the variance exists. Another possible convergent result is to analyze the convergence in expectation or high probability under a stochastic starting point. However our divergent result holds for any initial point, which rules out this possibility.

Without relying on the mini-batch size, we make a direct analysis of variance and the convergence of ADAM. We first show a motivating result which points out that the convergence of an ADAM-type algorithm can be implied by reducing the variance. Motivated by this, we propose a variance reduced version of ADAM, called VRADAM, and show that VRADAM converges. We provide two options regarding to resetting of ADAM states during the full gradient steps, and recommend the resetting option based on a theoretical analysis herein and computational experiments. Finally, we conduct several computational experiments, and show that our algorithm performs as well as the original version of ADAM.

In Section 3, we show a divergent example. Using contradiction by assuming the algorithm converges, we show that the expected update of iterates is larger than a positive constant, which means that it is impossible for the algorithm to converge to an optimal solution, which contradicts with the assumption. In Section 5, we prove the convergence of VRADAM. The main proof technique applied is to properly bound the difference between the estimated gradients and the true value of gradients. By bounding the update of the objective function in each iterate, we can further employ the strong convexity assumption and conclude convergence.

Our contributions are as follows.

1. We provide an unconstrained and strongly convex stochastic optimization problem on which the original ADAM diverges. We show that the divergence holds for any initial point, which rules out all of the possible weaker convergent results under stochastic starting point.

2. We construct a divergent mini-batch problem with $b = N - 1$, and conclude that there does not exist a convergent threshold for the mini-batch size.

3. We propose a variance reduced version of ADAM. We provide convergence results of the variance reduced version for strongly convex objectives to optimality or non-convex objectives. We show by experiments that the variance reduction does not harm the numerical performance of ADAM.

In Section 2, we review the literature on the topics of the convergence/divergence issue of ADAM and variance reduction optimization methods. In Section 3 we provide divergent examples for stochastic ADAM. We show that the example is divergent for large batch sizes, which disproves the existence of a convergence threshold of mini-batch size. In Section 4 we start from a reducing variance condition and prove the convergence of an ADAM-type algorithm under this condition. In Section 5 we propose a variance reduced version of ADAM. We show that resetting the states in the algorithm helps with the performance. We also provide a convergence result of our variance reduced ADAM. In Section 6 we conduct several numerical experiments and show the convergence and sensitivity of the proposed algorithm.

2 Literature Review

Convergence of ADAM: Reddi et al. [Reddi et al., 2018] firstly pointed out the convergence issue of ADAM and proposed two convergent variants: (a) AMSGrad takes the historical maximum value of v_t to keep the step size decreasing and (b) ADAMNC requires the hyper-parameters to satisfy specific conditions. Both of the approaches require that β_1 varies with time, which is inconsistent with practice. Fang and Klabjan [Fang and Klabjan, 2019] gave a convergence proof for AMSGrad with constant β_1 and [Alacaoglu et al., 2020] provided a tighter bound. Enlarging the mini-batch size is another direction. [De et al., 2018] and [Shi et al., 2020] proved the convergence of ADAM for full batch gradients and [Zaheer et al., 2018] showed the convergence of ADAM as long as the batch size is of the same order as the maximum number of iterations, but one criticism is that such a setting for the batch size is very inefficient in practice since the calculation of a large batch gradient is expensive. Several works, such as [Guo et al., 2021, Zou et al., 2019, Wang et al., 2019] proposed guidelines on setting hyper-parameters in order to obtain convergent results. [Guo et al., 2021] showed that as long as β_1 is close enough to 1, in particular, $1 - \beta_{1,t} \propto 1/\sqrt{t}$, ADAM establishes a convergent rate of $\mathcal{O}(1/\sqrt{T})$. However, since [Reddi et al., 2018] proposed the divergent example for any fixed β_1 and β_2 such that $\beta_1 < \sqrt{\beta_2}$, there is no hope to extend the results of [Guo et al., 2021] to constant momentum parameters. [Zou et al., 2019] also provided a series of conditions under which ADAM could converge. Specifically, they require the quantity $\alpha_t/\sqrt{1-\beta_{2,t}}$ to be 'almost' non-increasing. [Wang et al., 2019] proposed to set the denominator hyper-parameter ϵ to be $1/t$, and showed the convergence of ADAM for strongly convex objectives. The aforementioned works focus on setting the hyper-parameters in ADAM. On contrary, our work proposes a new algorithm that only requires basic and common conditions. We show a $\mathcal{O}(T^{-p})$ convergence rate for $0 < p < 1$ where p is dependent on hyper-parameters.

Variance reduction: The computational efficiency issues of full gradient descent methods get more severe with a large data size, but employing stochastic gradient descent may cause divergence because of the issue of variance. One classic method for variance reduction is to use mini-batch losses with a larger batch size, which however does not guarantee the variance to converge to zero. As an estimation of the full gradient, the stochastic average gradient (SAG) method [Le Roux et al., 2012] uses an average of $\nabla f_i(x_{k_i})$, where k_i is the most recent step index when sample i is picked. Although the convergence analysis of SAG provided in [Schmidt et al., 2017] showed its remarkable linear convergence, the estimator of the descent direction is biased and the analysis of SAG is complicated. SAGA [Defazio et al., 2014], an unbiased variant to SAG introduced a concept called 'covariates' and guarantees linear convergence as well. Both SAG and SAGA require the memory of $\mathcal{O}(Nd)$, which is expensive when the data set is large. SVRG [Johnson and Zhang, 2013] constructs two layers of iterations and calculates the full gradient as an auxiliary vector for variance reduction before starting each inner loop. It only requires a memory of $\mathcal{O}(d)$. Most of the literature on variance reduction focus on the convergence rate and memory requirement on the plain SGD algorithm. Recently, [Dubois-Taine et al., 2021] combined AdaGrad with SVRG for robustness in the learning rate. Our work introduces the idea of variance reduction to the convergence analysis of ADAM. It initiates the idea of the dynamic learning rate to SVRG.

3 Divergent examples for stochastic ADAM with large batch size

Several recent works [Shi et al., 2020, Zaheer et al., 2018, De et al., 2018] have suggested increasing the mini-batch size may help with convergence of ADAM. In particular, vanilla ADAM is convergent if the mini-batch size b is equal to the size of the training set, or it increases in the same order as training iterates. An interesting question is whether there exists a threshold of the batch size $b^* = b(N)$, which is smaller than N, such that $b > b^*$ implies convergence of ADAM. If such a threshold exists, the convergence can be guaranteed by a sufficiently large, but neither increasing nor as large as the training set size, batch size. Unfortunately, such a threshold does not exist. In fact, we show in this section that as long as the algorithm is not full batch, one can find a divergent example of ADAM.

Another aspect of interest is if ADAM converges on average or with high probability. Our example establishes non-convergence for any initial data point (even starting with an optimal one). We conclude that a probabilistic statement is impossible if stochasticity comes from either sampling or the initial point.

Reddi et al. firstly proposed a divergent example for ADAM in [Reddi et al., 2018]. The example, which is under the population loss minimization framework, consists of two linear functions defined on a finite interval. One drawback of this example is that the optimization problem is constrained, yet training in machine learning is usually an unconstrained problem. Under the unconstrained framework, the example proposed in [Reddi et al., 2018] does not have a minimum solution, hence it does not satisfy the basic requirements. We firstly propose an unconstrained problem under the population loss minimization framework.

Let a random variable ξ take discrete value from the set $\{1, 2\}$, and set $\mathbb{P}(\xi = 1) = \frac{1+\delta}{1+\delta^4}$ for some $\delta > 1$. Furthermore, we define the estimation of gradients by

$$\mathcal{G}(w; 1) = \frac{w}{\delta} + \delta^4 \quad \text{and} \quad \mathcal{G}(w; 2) = \frac{w}{\delta} - 1,$$

which implies the stochastic optimization problem with the loss functions

$$f_1(w) = \frac{w^2}{2\delta} + \delta^4 w \text{ and } f_2(w) = \frac{w^2}{2\delta} - w$$

with the corresponded probability distribution with respect to ξ. The population loss is given as $F(w) = \mathbb{E}_\xi [f_\xi(w)]$. We call this stochastic optimization problem the **Original Problem**(δ), or **OP**(δ) for short. We should note that OP(δ) is defined on $\mathbb{R}$, thus it is unconstrained. In addition, it is a strongly convex problem. As a divergent property of OP(δ), we show the following result.

Theorem 1 *There exists a $\delta^* > 2$ such that for any $\delta > \delta^*$ and any initial point w_1, ADAM diverges in expectation on OP(δ), i.e., $\mathbb{E}[F(w_t)] \not\to F^*$ where F^* is the optimal value of $F(w)$.*

The proof of Theorem1 is given in the appendix, where we show that for large enough δ, the expectation of the ADAM update between two consequential iterates is always positive. As a consequence, the iterates keep drifting from the optimal solution. The divergent example also tells us that strong convexity and the relaxation of constraints cannot help with the convergence of ADAM.

Based on the construction of OP(δ), we can give the divergent examples for any fixed mini-batch size.

Theorem 2 *For any fixed b, there exists an N_b^*, such that for any $N > N_b^*$, there exists a mini-batch problem with sample size N and batch size b where ADAM diverges for any initial point.*

Even if the batch size is unreasonably large, say $b = N - 1$, we can still construct the divergent example based on OP(δ) as stated next.

Theorem 3 *There exists an N^* such that for any $N > N^*$, there exists a mini-batch problem with sample size N and batch size $b = N - 1$ where ADAM diverges for any initial point.*

In conclusion, Theorem 2 and Theorem 3 extinguish the hope of finding a large enough batch size for stochastic ADAM to converge. Among the related works regarding the convergence of ADAM and batch size, larger batch size is always suggested, but the results in this section have enlightened the limitations of such approaches.

4 Motivation

In this section, we stick with the general ADAM algorithm described in Algorithm 1. To analyze, we make several assumptions on the gradient estimator and objective.

Assumption 1 *The gradient estimator $\mathcal{G} : \mathbb{R}^d \times \Omega \to \mathbb{R}^d$ and objective $F : \mathbb{R}^d \to \mathbb{R}$ satisfy the following:*

Algorithm 1 General ADAM

```
Require: Gradient estimation $\mathcal{G}(\cdot;\cdot)$, seed generation rule $\mathbb{P}_\xi$, initial point $w_1$, mini-batch size $b$, learning rate $\alpha_t$, exponential decay rates $\beta_1, \beta_2 \in [0, 1)$, denominator hyperparameter $\epsilon > 0$.
$m_0 \leftarrow 0, v_0 \leftarrow 0$
for $t \in 1, \dots T$ do
    Sample $\xi_t \sim \mathbb{P}_\xi$
    $g_t \leftarrow \mathcal{G}(w_t; \xi_t)$
    $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$
    $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_t \odot g_t$
    $V_t \leftarrow \mathrm{diag}(v_t) + \epsilon I_d$
    $w_{t+1} \leftarrow w_t - \alpha_t V_t^{-1/2} m_t$
end for
```

1. *$\mathcal{G}$ is unbiased, i.e., for any $w \in \mathbb{R}^d$, $\mathbb{E}_\xi [\mathcal{G}(w; \xi)] = \nabla F(w)$.*

2. *There exists a constant $0 < L < +\infty$, such that for any $\xi \in \Omega$ and $w, \bar{w} \in \mathbb{R}^d$, we have $\|\mathcal{G}(w; \xi) - \mathcal{G}(\bar{w}; \xi)\|_2 \le L\|w - \bar{w}\|_2$ and $\|\nabla F(w) - \nabla F(\bar{w})\|_2 \le L\|w - \bar{w}\|_2$.*

3. *There exists a constant $0 < G < +\infty$, such that for any $\xi \in \Omega$ and $w \in \mathbb{R}^d$, we have $\|\mathcal{G}(w; \xi)\|_2 \le G$ and $\|F(w)\|_2 \le G$.*

As this point convexity is not needed. We mainly focus on the variance of the gradient estimator. The common assumptions in the literature are that the variance is bounded by a constant, [Zaheer et al., 2018], or a linear function of the square of the norm of the objective $\mathrm{Var}(\mathcal{G}_i(w; \xi)) \le C_1 + C_2\|\nabla F(w)\|_2^2$, [Bottou et al., 2018]. Another assumption made in [Shi et al., 2020, Vaswani et al., 2019] is called the 'strongly growth condition' which is $\sum_{n=1}^N \|\nabla f_n(w)\|_2^2 \le C\|\nabla F(w)\|_2^2$ for some $C > 0$. Note that for vanilla ADAM where $\mathcal{G}(w; \xi) = \nabla F^{\mathcal{B}}(w)$ the strongly growth condition implies that $\nabla F^{\mathcal{B}}(w^*) = 0$ if and only if $\nabla F(w^*) = 0$. As a result the strongly growth condition implies that $\mathrm{Var}(\mathcal{G}(w; \xi)) \le 2L\mathbb{E}[\|w - w^*\|_2^2]$, given Lipshitz smooth gradients for full-batch and mini-batch losses. For those iterates close to a saddle point, the variance is automatically reduced, because $\|w - w^*\|_2^2$ is small. However, the strongly growth condition is so strong that the majority of practical problems do not satisfy it. In fact, one observation of OP(δ) is that the variance is a constant, which also breaks the strongly growth condition.

In this section, as a motivative result, let us assume the variance of the gradient estimator is reduced a priori. Let us denote a series of positive constants $\{\lambda_t\}_{t=1}^T$ such that for any $t = 1, \dots, T$, we have $\mathrm{Var}(\mathcal{G}(w_t; \xi_t)) \le \lambda_t$. For the objective with a finite lower bound, we have the following result.

Theorem 4 *Let Assumption 1 be satisfied, and assume that*

$F(w)$ is lower bounded by $F_{\inf} > -\infty$. Then for any initial point w_1, ADAM satisfies

$$\min_{1\le t\le T} \mathbb{E}\left[\|\nabla F(w_t)\|_2^2\right] \le \mathcal{O}\left(\frac{\sum_{t=1}^T \alpha_t^2}{\sum_{t=1}^T \alpha_t} + \frac{\sum_{t=1}^T \alpha_t\lambda_t}{\sum_{t=1}^T \alpha_t}\right).$$

The proof is in the appendix. Let us assume that the two common conditions $\sum_{t=1}^\infty \alpha_t = \infty$ and $\sum_{t=1}^\infty \alpha_t^2 < \infty$ are satisfied. Theorem 4 shows that ADAM converges if $\sum_{t=1}^\infty \alpha_t\lambda_t < +\infty$. In fact, $\lambda_t \to 0$ as $t \to \infty$ implies that $\sum_{t=1}^T \alpha_t\lambda_t / \sum_{t=1}^T \alpha_t \to 0$, and hence it leads to convergence of the algorithm.

We emphasize that since the assumption on variance is made on the algorithmic iterates $\{w_t\}_{t=1}^T$, it is very difficult to be checked for a specific problem in advance. However, we showed that if the variance is convergent, an ADAM-type algorithm converges. We show next that the algorithm we propose has convergent variance and furthermore is convergent.

5 Variance Reduced ADAM

Algorithm 2 Variance Reduced ADAM

```
Require: Loss functions $\{f_n(w)\}_{n=1}^N$, initial point $\widetilde{w}_1$, learning rate $\alpha_t$, exponential decay rates $\beta_1, \beta_2 \in [0,1)$, denominator hyper-parameter $\epsilon > 0$, inner iteration size $m$. Initialize $m_m^{(0)} \leftarrow 0$, $v_m^{(0)} \leftarrow 0$.
  for $t = 1, \dots, T$ do
    Compute full-batch gradient $\nabla F(\widetilde{w}_t)$
    $w_1^{(t)} \leftarrow \tilde{w}_t$
    Option A: (Resetting) $m_0^{(t)} \leftarrow 0$, $v_0^{(t)} \leftarrow 0$
    Option B: (No Resetting) $m_0^{(t)} \leftarrow m_m^{(t-1)}$ and $v_0^{(t)} \leftarrow v_m^{t-1}$
    for $k = 1, \dots, m$ do
      Sample $\mathcal{B}_k^{(t)}$ from $\{1, \dots, N\}$ with $\left|\mathcal{B}_k^{(t)}\right| = b$.
      $g_k^{(t)} \leftarrow \nabla F^{\mathcal{B}_k^{(t)}}\left(w_k^{(t)}\right) - \nabla F^{\mathcal{B}_k^{(t)}}(\widetilde{w}_t) + \nabla F(\widetilde{w}_t)$
      $m_k^{(t)} \leftarrow \beta_1 m_{k-1}^{(t)} + (1-\beta_1) g_k^{(t)}$
      $v_k^{(t)} \leftarrow \beta_2 v_{k-1}^{(t)} + (1-\beta_2) g_k^{(t)} \odot g_k^{(t)}$
      Option A: $\widetilde{m}_k^{(t)} \leftarrow \frac{m_k^{(t)}}{1-\beta_1^k}$, $\widetilde{v}_k^{(t)} \leftarrow \frac{v_k^{(t)}}{1-\beta_2^k}$
      Option B: $\widetilde{m}_k^{(t)} \leftarrow \frac{m_k^{(t)}}{1-\beta_1^{k+(t-1)m}}$, $\widetilde{v}_k^{(t)} \leftarrow \frac{v_k^{(t)}}{1-\beta_2^{k+(t-1)m}}$
      $V_k^{(t)} \leftarrow \mathrm{diag}\left(\widetilde{v}_k^{(t)} + \epsilon\right)$
      $w_{k+1}^{(t)} \leftarrow w_k^{(t)} - \alpha_t \left(V_k^{(t)}\right)^{-1/2} \widetilde{m}_k^{(t)}$
    end for
    $\widetilde{w}_{t+1} \leftarrow w_{m+1}^{(t)}$
  end for
```

Variance reduction for random variables is a common topic in many fields. In general, an unbiased variance reduction of a random variable X is $\tilde{X} = X - Y + \mathbb{E}Y$, which establishes the variance $\mathrm{Var}(\tilde{X}) = \mathrm{Var}(X) + \mathrm{Var}(Y) - 2\mathrm{Cov}(X,Y) < \mathrm{Var}(X)$ given $\mathrm{Cov}(X,Y) > \mathrm{Var}(Y)/2$, i.e., X and Y are positively correlated at a sufficient level. In the context of stochastic gradient descent, the random variable for variance reduction is $\mathcal{G}(w_t;\xi_t)$, the gradient of mini-batch loss $\nabla F^{\mathcal{B}_t}(w_t)$. Johnson and Zhang [Johnson and Zhang, 2013] proposed a solution for SGD. They suggested the associate random variable to be the gradient of the same mini-batch loss at a previous iterate $\tilde{w}$. Since the expectation of a mini-batch gradient is the full-batch gradient, the descent direction becomes $g_t = \nabla F^{\mathcal{B}_t}(w_t) - \nabla F^{\mathcal{B}_t}(\tilde{w}) + \nabla F(\tilde{w})$. Vector $\tilde{w}$ is known as the snapshot model. Since calculation of the full batch gradient at $\tilde{w}$ is required, [Johnson and Zhang, 2013] proposed to save the snapshot model every m iterations, which is known as the SVRG algorithm. Inspired by SVRG and motivated by the result in Section 4, we propose the combination of the variance reduce method and ADAM, called VRADAM (Algorithm 2).

An intuitive analysis of variance of the update direction

$$\begin{aligned}
\mathrm{Var}\left(g_{k,i}^{(t)}\right) &= \mathrm{Var}\left(\nabla_i F^{\mathcal{B}_k^{(t)}}\left(w_k^{(t)}\right) - \nabla_i F^{\mathcal{B}_k^{(t)}}(\tilde{w}_t)\right) \\
&\le \mathbb{E}\left[\left(\nabla_i F^{\mathcal{B}_k^{(t)}}\left(w_k^{(t)}\right) - \nabla_i F^{\mathcal{B}_k^{(t)}}(\tilde{w}_t)\right)^2\right] \\
&\le L^2 \mathbb{E}\left[\left\|w_k^{(t)} - \tilde{w}_t\right\|_2^2\right].
\end{aligned}$$

is that as the iterates become close to the optimal point, the variance is reduced simultaneously, which guarantees a similar condition of variance as the strongly growth condition.

5.1 Resetting/No resetting options

We provide two options with regard to the update of ADAM states. In one option, we reinitialize the ADAM states at the beginning of each outer iteration, while the other option keeps the state through the whole training process. Although for the original ADAM, resetting the states harms the performance of the algorithm, we computationally found that the resetting option works better in VRADAM. Intuitively, this is because in each inner loop, the first step $g_1^{(t)}$ is always the full gradient direction, which makes a more efficient update than the direction adapted by previous ADAM states. In order to support our argument, we provide a theoretical analysis of an example. If we fix the initial point $w_1 \in \mathbb{R}$ and the mini-batch losses $F^{\mathcal{B}_1^{(1)}}, F^{\mathcal{B}_2^{(1)}}, \dots, F^{\mathcal{B}_m^{(1)}}$, the iterates are identical between the two options through the $t = 1$ iteration. We consider the objective values after the first update in the second outer iteration, i.e. $F\left(w_2^{(2)}\right)$. At the end of $t = 1$ iteration, we obtain $w_{m+1}^{(1)} = w_1^{(2)} = \tilde{w}_2$ and the ADAM states $m_{m+1}^{(1)}$ and $v_{m+1}^{(1)}$. Then while $t = 2$, the algorithm makes the first update as follows.

Option A:

$$m_1^{(2)} = (1-\beta_1)g_1^{(2)}$$
$$\widetilde{m}_1^{(2)} = g_1^{(2)}$$
$$v_1^{(2)} = (1-\beta_2)\left(g_1^{(2)}\right)^2$$
$$\widetilde{v}_1^{(2)} = \left(g_1^{(2)}\right)^2$$
$$w_2^{(2)} = w_1^{(2)} - \alpha_2 \frac{\widetilde{m}_1^{(2)}}{\sqrt{\widetilde{v}_1^{(2)}} + \epsilon}$$

Option B:

$$\hat{m}_1^{(2)} = \beta_1 m_{m+1}^{(1)} + (1-\beta_1)g_1^{(2)}$$
$$\hat{\widetilde{m}}_1^{(2)} = \hat{m}_1^{(2)}/(1-\beta_1^{m+1})$$
$$\hat{v}_1^{(2)} = \beta_2 v_{m+1}^{(1)} + (1-\beta_2)\left(g_1^{(2)}\right)^2$$
$$\hat{\widetilde{v}}_1^{(2)} = \hat{v}_1^{(2)}/(1-\beta_2^{m+1})$$
$$\hat{w}_2^{(2)} = w_1^{(2)} - \alpha_2 \frac{\hat{\widetilde{m}}_1^{(2)}}{\sqrt{\hat{\widetilde{v}}_1^{(2)}} + \epsilon}$$

We make the following assumptions.

Assumption 2 *The framework described in this section satisfies:*

1. *$F(w)$ is $c-$strongly convex, and its gradient is $L-$smooth. Each one of $\left|g_1^{(1)}\right|, \dots, \left|g_m^{(1)}\right|$ and $\left|g_1^{(2)}\right|$ is bounded above by $G > 0$.*
2. *The algorithm makes progress in the $t = 1$ iteration, specifically, $\left|m_{m+1}^{(1)}\right| \geq \left|F'\left(w_1^{(2)}\right)\right|$.*
3. *The hyper-parameters satisfy*

$$L\alpha_2 \geq 2\sqrt{G^2+\epsilon} \text{ and } \frac{L}{c} \leq \frac{2\beta_1 - 1}{1-\beta_1^{m+1}}\sqrt{\frac{\epsilon}{G^2+\epsilon}}.$$

Notice that the second assumption assumes that the exponential moving average of the steps in the first loop is larger than the full gradient at the beginning of the second loop, which reflects that the algorithm makes progress in the first iteration. The third assumption can be satisfied by β_1 that is close enough to 1 and appropriately selected α_2. Our concern is to compare the objective values $F\left(w_2^{(2)}\right)$ and $F\left(\hat{w}_2^{(2)}\right)$ of the two options. The following theorem shows that option A, i.e., the option where the states of ADAM are reset at the beginning of each outer iteration, makes more efficient descent, hence works better.

Theorem 5 *Given Assumption 2, we have $F\left(\hat{w}_2^{(2)}\right) \geq F\left(w_2^{(2)}\right)$.*

While Theorem 5 allows the preference of option A within the two outer iterations, the computational experiments confirm this choice in general.

5.2 Convergence results for VRADAM with resetting

In the previous section we show the advantage of resetting of the ADAM states every outer iteration over not doing so by comparing the values of the objectives in the two options. In this section, we provide a convergence proof of the resetting option of VRADAM. Similar to Assumption 1 we make under the general ADAM framework, we make the following specific assumptions for the convergence proof of VRADAM.

Assumption 3 *The loss functions $f_1(w), \dots, f_N(w)$ satisfy the following conditions.*

1. *There exists a constant $0 < L < +\infty$, such that for any $n = 1, \dots, N$ and $w, \bar{w} \in \mathbb{R}^d$, we have $\|\nabla f_n(w) - \nabla f_n(\bar{w})\|_2 \leq L\|w - \bar{w}\|_2$.*
2. *There exists a constant $0 < G < +\infty$, such that for any $n = 1, \dots, N$ and $w \in \mathbb{R}^d$, we have $\|\nabla f_n(w)\|_2 \leq G$.*

We show the convergence result of VRADAM for strongly convex functions first.

Theorem 6 *Let Assumption 3 be satisfied and assume that $F(w)$ is strongly convex with parameter c, and let F^* be the unique minimum of F. Let $\alpha_t = \alpha/t$ and we require $C_2 = 2c(1-\beta_1)/\sqrt{9G^2+\epsilon} < 1/\alpha m$. Then for any initial point w_1, Algorithm 2 with Option A satisfies $F(\widetilde{w}_T) - F^* \leq \mathcal{O}\left(T^{-C_2 m\alpha}\right)$ almost surely.*

We remark that the requirement $C_2 m\alpha < 1$ can be satisfied by properly selected α and β_1. Specifically, when β_1 is close to 1 and α is small, the assumption is more likely to be satisfied. The proof starts by bounding the update of the objective function in one iterate and then applies strong convexity.

As for objectives that are not necessarily convex, we exhibit the following result.

Theorem 7 *Let Assumption 3 be satisfied and assume that $F(w)$ is lower bounded by $F_{\inf} > -\infty$. We require $\sum_{t=1}^{\infty}\alpha_t^2 < +\infty$ and $\sum_{t=1}^{\infty}\alpha_t = +\infty$. Then for any initial point w_1, Algorithm 2 with Option A satisfies $\liminf_{t\to\infty}\|\nabla F(\tilde{w}_t)\|_2 = 0$ almost surely.*

Under the general objective setting, as a corollary of Theorem 7, we can bound the variance of the norm of the gradient.

Corollary 1 *Given the same conditions as in Theorem 7 , we have $\liminf_{t\to\infty}\mathrm{Var}\left(\|\nabla F(\widetilde{w}_t)\|_2\right) = 0$.*

The proof of the corollary simply comes from

$$\liminf_{t\to\infty}\mathrm{Var}(\|\nabla F(\widetilde{w}_t)\|_2) \leq \liminf_{t\to\infty}\mathbb{E}(\|\nabla F(\widetilde{w}_t)\|_2^2)$$

where the right hand side is 0 because almost sure convergence implies L2 convergence.

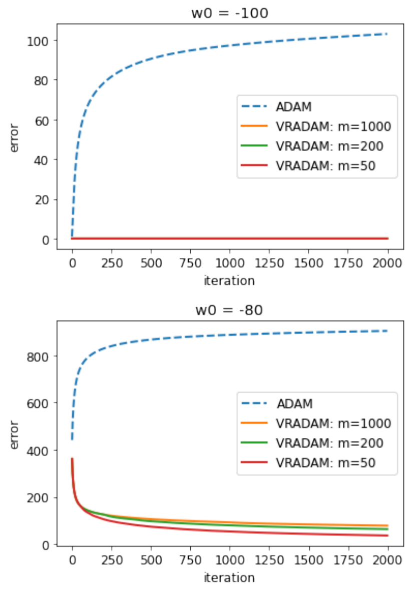

Figure 1: The numerical experiments of OP(δ)

6 Numerical Experiments

6.1 Divergent example

In Section 3, we provide an unconstrained stochastic optimization problem where ADAM diverges for any initial point. The goal of this section is to numerically show that ADAM diverges on this problem. Since the construction of the mini-batch problem can be equivalently transformed into a stochastic optimization problem with population loss, we stick to the experiments of OP(δ) defined in Section 3. Letting $\delta = 10$, the optimal solution of OP(δ) is $w^* = -\delta^2 = -100$. We consider two cases: when the algorithms start from the optimal solution , i.e., $w_0 = -100$ and when they start from somewhere far from the optimal solution, in the experiments we set $w_0 = -80$. We simulate 1,000 trials for each case and plot the expected L2 error $\mathbb{E}[(w_t - w^*)^2]$ of w. We notice from Figure1 that even starting from the optimal point, ADAM still diverges, while VRADAM converges well. When $w_0 = -80$, VRADAM eventually converges to the optimal solution while ADAM diverges. This result rules out all of the possible temptations of solving the divergence problem of ADAM by using a stochastic initial point.

6.2 Experiments on machine learning problems

6.2.1 Datasets and implementation

In this section, we compare the numerical performances of VRADAM and ADAM on several real-world classification tasks. The experiments are conducted on the following data sets.

- **Coverage Type** [Blackard et al., 1998]: A dataset predicting forest cover type from 54 cartographic variables. The dataset contains 581,012 data points and assigns them into 7 different categories.
- **MNIST** [Deng, 2012]: A handwritten digit dataset containing 60,000 grey level images with size 28×28 pixels.
- **NSL-KDD** [Tavallaee et al., 2009]: A selected subset of the KDD CUP 99 dataset, which is a public dataset used to train a network intrusion detection system. The subset eliminates repeated data samples and avoids several shortcomings in the original dataset.
- **Embedded CIFAR-10**: CIFAR-10 [Krizhevsky and Hinton, 2009] consists of 60,000 color images in 10 classes. We feed each sample to a pretrained ResNet model [He et al., 2016] and obtain a 1,000 dimensional embedding vector for each image.

We use logistic regression on the four previously mentioned data sets and non-convex deep neural networks on MNIST and Coverage Type datasets. The structures of the deep neural networks used in this section are explained in the appendix. Cross entropy is the underlying loss. While training these models, we fix the batch size $|\mathcal{B}_t| = 64$ and the ADAM hyper-parameters $\beta_1 = 0.9, \beta_2 = 0.999$, which are commonly recommended values in the literature and fine tune the learning rate schedules among $\alpha_t = \alpha_0$, $\alpha_t = \alpha_0/t$ and $\alpha_t = \alpha_0\gamma^t$, i.e., the constant learning rate, inverse-proportional learning rate and the exponentially decaying learning rate. We perform a grid search among $\alpha_0 \in \{0.0005, 0.001, 0.005, 0.01, 0.05\}$ and $\gamma \in \{0.6, 0.8, 0.95\}$.

In order to eliminate luck from randomness, we ran each experiment setting with 3 different random seeds. We report the average loss of each experiment. We train each setting for 15 epochs for VRADAM and 50 epochs for ADAM. While comparing the performances, we display the loss functions up to convergence. The experiments were conducted in PyTorch 1.12.1 on the Google Colab cloud service.

6.2.2 Main results

As for VRADAM, the number of inner loop iterations m in Algorithm 2 is also a hyper parameter to decide. If m is too large, the variance reduce procedure does not make a real difference, while an improperly small m would lead to a very frequent computation of the full gradients, which is computationally expensive. We recommend that the length of the inner loop should be about the size of an epoch. In other word, $m \approx N/b$, where N is the number of samples and b is the mini-batch size. In our experiments, we test the performance of VRADAM with $m \in \{N/2b, N/b, 2N/b, 4N/b\}$.

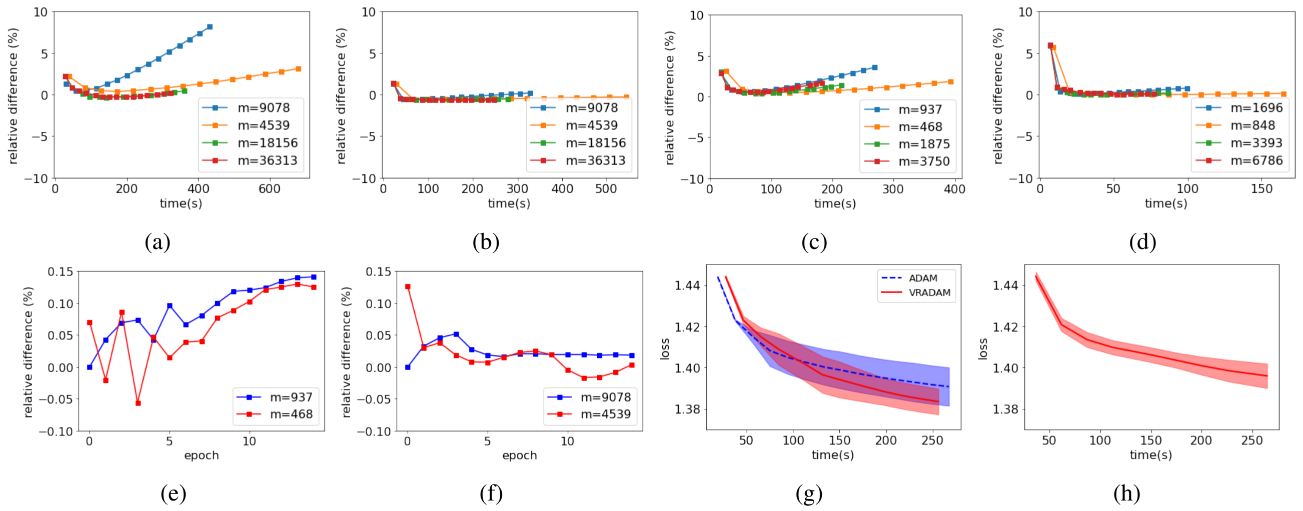

Figure 2: Top: Relative differences of loss function of VRADAM over ADAM on classification tasks of (a) CovType dataset with a feedforward network and (b) CovType with logistic regression (c) MNIST with logistic regression (d) NSL-KDD with logistic regression. Bottom left: Relative differences of loss function of VRADAM without reset over VRADAM with reset on classification tasks of CovType dataset with (e) a feedforward network and (f) logistic regression. Bottom right: Deviation of loss. The shaded areas mark the maximal and minimal losses among the three seeds.

It is unfair to compare the convergence of the training loss with regard to the number of epochs, since the time VRADAM spends on training one epoch is longer than ADAM. Instead, we compare the convergence rate with regard to the computational time.

Figures 2a, 2b, 2c and 2d show the relative difference of the loss of VRADAM with regard to ADAM. It shows that our approach of variance reduction has as good convergence rate as ADAM. Although VRADAM converges slower than ADAM in a few starting iterations, due to the first full gradient computation, the variance reduction approach can catch up and reach a similar convergence rate as ADAM. Specifically, we observe that VRADAM works better on large datasets, such as CovType and NSL-KDD. We also find that VRADAM works better on convex problems by comparing Figures 2a and 2b.

As we mentioned in Section 5, we recommend resetting the optimizer states every inner loop. Figures 2e and 2f display the performance of the resetting option in a few experiments and show that the resetting option helps with the convergence of VRADAM. Additional experiments are shown in the appendix.

In conclusion, we observe that VRADAM outperforms ADAM on large datasets, such as CovType. Such datasets may contain extreme values, which may harm the convergence of ADAM. We find that the computational cost when calculating the full gradients can be compensated by the benefits of quick convergence of VRADAM. We recommend VRADAM over ADAM for tasks with large datasets, in particular if loss is convex.

6.2.3 Sensitivity

We consider the CovType classification task with the FFN model as an example to analyze the sensitivity of our algorithm. As we stated previously, our algorithm fixes the convergence issue of ADAM by reducing the variance. Our experiments take three different seeds and the deviation of the results reflects the variance of the algorithm. Figure 2g shows that VRADAM reduces the noise comparing with ADAM. We also studied the sensitivity of VRADAM over the different initial points, which is shown in Figure 2h.

7 Conclusions

We started from an analysis of the divergence of the original ADAM algorithm and concluded that even strong convexity can not help with the convergence of ADAM. We then gave a convergence analysis under a high-level variance reduction assumption, and concluded that an ADAM-type algorithm converges if its variance is reduced. Inspired by this motivating result and the idea of SVRG, we proposed a variance reduced approach which fixes the original convergence issue of ADAM. We finally showed with numerical experiments that even though our approach requires more gradient computation than ADAM, VRADAM converges as quickly as ADAM after several initial iterations. The motivating results we provided can also lead to other variance reduction approaches, which are possible future research directions on the convergence issue of ADAM.

References

[Alacaoglu et al., 2020] Alacaoglu, A., Malitsky, Y., Mertikopoulos, P., and Cevher, V. (2020). A new regret analysis for ADAM-type algorithms. In *International Conference on Machine Learning*.

[Blackard et al., 1998] Blackard, J. A., Dean, D. J., and Anderson, C. W. (1998). UCI machine learning repository: Covertype data set. `https://archive.ics.uci.edu/ml/datasets/covertype`.

[Bottou et al., 2018] Bottou, L., Curtis, F. E., and Nocedal, J. (2018). Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311.

[De et al., 2018] De, S., Mukherjee, A., and Ullah, E. (2018). Convergence guarantees for RMSprop and ADAM in non-convex optimization and an empirical comparison to Nesterov acceleration. *arXiv preprint arXiv:1807.06766*.

[Defazio et al., 2014] Defazio, A., Bach, F., and Lacoste-Julien, S. (2014). SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in Neural Information Processing Systems*.

[Deng, 2012] Deng, L. (2012). The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142.

[Dubois-Taine et al., 2021] Dubois-Taine, B., Vaswani, S., Babanezhad, R., Schmidt, M., and Lacoste-Julien, S. (2021). SVRG meets AdaGrad: Painless variance reduction. *arXiv preprint arXiv:2102.09645*.

[Duchi et al., 2011] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7).

[Fang and Klabjan, 2019] Fang, B. and Klabjan, D. (2019). Convergence analyses of online adam algorithm in convex setting and two-layer relu neural network. *arXiv preprint arXiv:1905.09356*.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

[Guo et al., 2021] Guo, Z., Xu, Y., Yin, W., Jin, R., and Yang, T. (2021). A novel convergence analysis for algorithms of the ADAM family. *arXiv preprint arXiv:2112.03459*.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*.

[Hinton et al., 2012] Hinton, G., Srivastava, N., and Swersky, K. (2012). Neural networks for machine learning: lecture 6a overview of mini-batch gradient descent. `https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf`.

[Johnson and Zhang, 2013] Johnson, R. and Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. *Advances in Neural Information Processing Systems*.

[Kingma and Ba, 2015] Kingma, D. P. and Ba, J. (2015). ADAM: A method for stochastic optimization. *International Conference on Learning Representations*.

[Krizhevsky and Hinton, 2009] Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. `https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.222.9220&rep=rep1&type=pdf`.

[Le Roux et al., 2012] Le Roux, N., Schmidt, M., and Bach, F. (2012). A stochastic gradient method with an exponential convergence rate for finite training sets. *Advances in Neural Information Processing Systems*.

[Qian and Klabjan, 2020] Qian, X. and Klabjan, D. (2020). The impact of the mini-batch size on the variance of gradients in stochastic gradient descent. *arXiv preprint arXiv:2004.13146*.

[Reddi et al., 2018] Reddi, S. J., Kale, S., and Kumar, S. (2018). On the convergence of ADAM and beyond. *International Conference on Learning Representations*.

[Schmidt et al., 2017] Schmidt, M., Le Roux, N., and Bach, F. (2017). Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1):83–112.

[Shi et al., 2020] Shi, N., Li, D., Hong, M., and Sun, R. (2020). RMSprop converges with proper hyper-parameter. In *International Conference on Learning Representations*.

[Tavallaee et al., 2009] Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A. (2009). A detailed analysis of the KDD cup 99 data set. In *2009 IEEE symposium on Computational Intelligence for Security and Defense Applications*.

[Vaswani et al., 2019] Vaswani, S., Bach, F., and Schmidt, M. (2019). Fast and faster convergence of SGD for over-parameterized models and an accelerated perceptron. In *The 22nd International Conference on Artificial Intelligence and Statistics*.

[Wang et al., 2019] Wang, G., Lu, S., Tu, W., and Zhang, L. (2019). SADAM: A variant of ADAM for strongly convex functions. *arXiv preprint arXiv:1905.02957*.

[Zaheer et al., 2018] Zaheer, M., Reddi, S., Sachan, D., Kale, S., and Kumar, S. (2018). Adaptive methods for nonconvex optimization. *Advances in Neural Information Processing Systems*.

[Zhou et al., 2019] Zhou, Z., Zhang, Q., Lu, G., Wang, H., Zhang, W., and Yu, Y. (2019). ADAshift: Decorrelation and convergence of adaptive learning rate methods. *International Conference on Learning Representations*.

[Zou et al., 2019] Zou, F., Shen, L., Jie, Z., Zhang, W., and Liu, W. (2019). A sufficient condition for convergences of ADAM and RMSprop. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.