AdaBlock: SGD with Practical Block Diagonal Matrix Adaptation for Deep Learning

Jihun Yun
KAIST

Aurelie C. Lozano
IBM T.J. Watson Research Center

Eunho Yang
KAIST, AITRICS

Abstract

We introduce AdaBlock, a class of adaptive gradient methods that extends popular approaches such as Adam by adopting the simple and natural idea of using block-diagonal matrix adaption to effectively utilize structural characteristics of deep learning architectures. Unlike other quadratic or block-diagonal approaches, AdaBlock has complete freedom to select block-diagonal groups, providing a wider trade-off applicable even to extremely high-dimensional problems. We provide convergence and generalization error bounds for AdaBlock, and study both theoretically and empirically the impact of the block size on the bounds and advantages over usual diagonal approaches. In addition, we propose a randomized layer-wise variant of Adablock to further reduce computations and memory footprint, and devise an efficient spectrum-clipping scheme for AdaBlock to benefit from Sgd's superior generalization performance. Extensive experiments on several deep learning tasks demonstrate the benefits of block diagonal adaptation compared to adaptive diagonal methods, vanilla Sgd, as well as modified versions of full-matrix adaptation.

1 Introduction

Stochastic gradient descent (Sgd, Robbins and Monro (1951)) is a dominant approach for training large-scale machine learning models such as deep networks. At each iteration of this iterative method, the model parameters are updated in the opposite direction of the gradient of the objective function typically evaluated on a mini-batch, with step size controlled by a learning rate. While vanilla Sgd uses a common learning rate across coordinates (possibly varying across time), several adaptive learning rate algorithms have been developed that scale the gradient coordinates by square roots of some form of average of the squared values of past gradients coordinates. The first key approach in this class, Adagrad Duchi et al. (2011); McMahan and Streeter (2010), uses a per-coordinate learning rate based on squared past gradients, and has been found to outperform vanilla Sgd on sparse data. However, in non-convex dense settings where gradients are dense, performance is degraded, since the learning rate shrinks too rapidly due to the accumulation of all past squared gradient in its denominator. To address this issue, variants of Adagrad have been proposed that use the exponential moving average (EMA) of past squared gradients to essentially restrict the window of accumulated gradients to only few recent ones. Examples of such methods include Adadelta Zeiler (2012), RMSprop Tieleman and Hinton (2012), Adam Kingma and Ba (2015), and Nadam Dozat (2016).

Despite their popularity and great success in some applications, the above EMA-based adaptive approaches have raised several concerns. Wilson et al. (2017) studied their out-of-sample generalization and observed that on several popular deep learning models their generalization is worse than vanilla Sgd. Recently, Reddi et al. (2018) showed that they may not converge to the optimum (or critical point) even in simple convex settings with constant minibatch size, and noted that the effective learning rate of EMA methods can increase fairly quickly while for convergence it should decrease or at least have a controlled increase over iterations. AMSGrad, proposed in Reddi et al. (2018) to fix this issue, did not yield conclusive improvements in terms of generalization ability. To simultaneously benefit from the generalization ability of vanilla Sgd and the fast training of adaptive approaches, Luo et al. (2019) recently proposed AdaBound and AMSBound as variants of Adam and AMSGrad, which employ dynamic bounds on learning rates to guard against

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151. Copyright 2022 by the author(s).

extreme learning rates. Chen et al. (2019) introduced AdaFom that only add momentum to the first moment estimate while using the same second moment estimate as AdaGrad. Zaheer et al. (2018) showed that increasing minibatch sizes enables convergence of Adam, and proposed Yogi which employs additive adaptive updates to prevent informative gradients from being forgotten too quickly. Yu et al. (2017) considered a variant of diagonal adaptation where, for each neural network layer, the gradients are normalized by the ℓ_2 norm of the layer's gradients.

We note that all the aforementioned adaptive algorithms deal with adaptation in a restricted way, namely they only employ *diagonal* information about Gradient of Outer-Product ($g_t g_t^\top$ where g_t is the stochastic gradient at time t, a.k.a. GOP).

Though initially discussed in Duchi et al. (2011), *full* matrix adaptation has been mostly ignored due to its prohibitive computations in high-dimensions. To alleviate the overhead, several approximations have been studied. Specifically, KFAC Martens and Grosse (2015) and Shampoo Gupta et al. (2018) approximate the curvature via Kronecker product, while TONGA Le Roux et al. (2007) and GGT Agarwal et al. (2019) reduce the dimensions of gradient outer-product, which is a component of the curvature, via relatively lower-dimensional gradient inner-product.

However, the aforementioned approaches are suboptimal in terms of computations and memory. Indeed in KFAC and Shampoo, the inversion of each Kronecker factor is still burdensome for large-scale deep learning tasks. An additional limitation of these approaches is that they can only encourage a layer-wise block diagonal structure. Similarly, TONGA and GGT require memory tens of times the parameter dimension, which is not appropriate for large-scale deep models.

Contributions. In this paper, we study an extended form of Sgd learning with *block-diagonal* matrix adaptation that can effectively utilize the structural characteristics of deep learning architectures. Specifically, we consider a simple yet effective strategy for gradient outer-product via coordinate grouping, which leads to a Sgd framework we call AdaBlock. Unlike other block-diagonal approaches, AdaBlock allows for complete freedom in selecting block-diagonal groups, providing a wider trade-off applicable even to extremely high-dimensional problems.The goal of this framework is to take advantage of richer information on interactions across different gradient coordinates, while relaxing the expensive computational cost of full matrix adaptation in large-scale problems. For this purpose, we introduce several grouping strategies that are practically useful in deep learning. We study AdaBlock framework theoretically and empirically, and the make the following contributions:

- We analyse the convergence of AdaBlock in the non-convex setting, uniform stability and generalization error, and provide theoretical insights on the benefits of using blocks. Our work is the *first* study to investigate how the block size affects convergence and generalization, both in theory and practice.
- We propose *spectrum-clipping*, a non-trivial extension of Luo et al. (2019) to further boost generalization by allowing the block diagonal matrix to become a constant multiple of the identity matrix in the latter part of training, as in vanilla Sgd.
- We propose a Randomized Adablock variant (RadaBlock) for faster per-iteration time and smaller memory footprint.
- We evaluate the training and generalization ability of our approaches on popular deep learning tasks. Our extensive experiments reveal that in terms of generalization block diagonal methods outperform diagonal approaches and several baselines such as vanilla SGD/KFAC/Shampoo/GGT even for small grouping sizes while remaining practical in terms of computations and memory footprint.

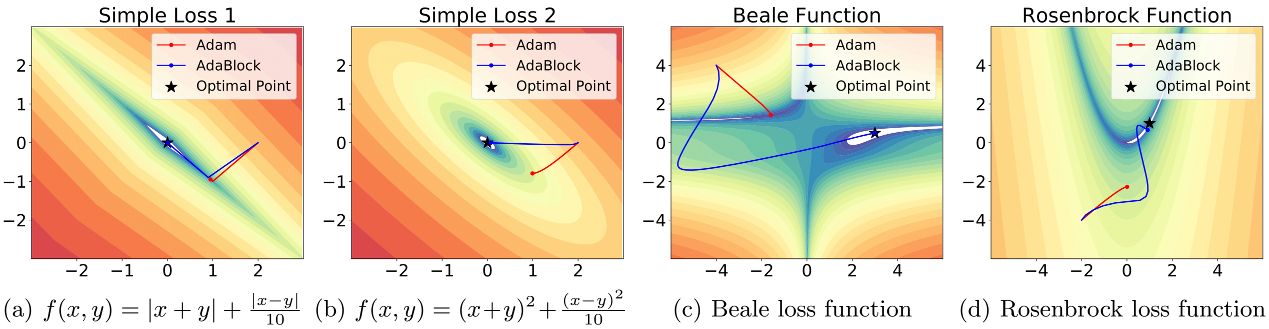
Notation. For a vector x, $\|x\|_p$ is the p-norm, and $\|x\|$ is $\|x\|_2$ if not specified. For a matrix A, $\|A\|_p$ is the matrix p-norm, $\lambda(A)$ returns eigenvalues (spectrum) of A, and1 $\log|A|$ denotes the log-determinant. $\lambda_{\min}(A)/\lambda_{\max}(A)$ denote the minimum/maximum eigenvalue of A respectively. $\mathrm{Clip}(x, a, b)$ means clipping x element-wise with the interval $I = [a, b]$.

2 Block-Diagonal Matrix Adaptation via Coordinate Partitioning

In the context of stochastic optimization, Duchi et al. (2011) proposed a full-matrix variant of AdaGrad. This version employs a preconditioner which exploits first-order information only, via the sum of outer products of past gradients:

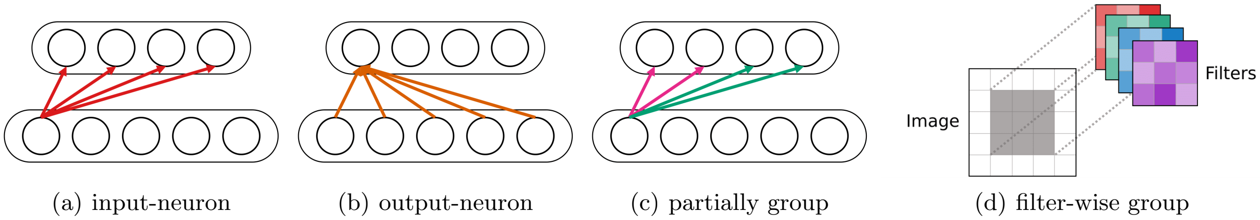
$$g_t = \nabla f(x_t), \quad G_t = G_{t-1} + g_t g_t^\top,$$
$$x_{t+1} = x_t - \alpha_t (G_t^{1/2} + \delta I)^{-1} g_t \tag{1}$$

where g_t is a stochastic gradient at time t, α_t is a step-size, and δ is a small constant for numerical stability. Duchi et al. (2011) presented regret bounds for (1) in the convex setting. However, this approach is quite expensive due to $G_t^{1/2}$ term, so they proposed to only use the diagonal entries of G_t. Popular adaptive methods for training deep models such as RMSprop/Adam



(a) $f(x,y) = |x+y| + \frac{|x-y|}{10}$ (b) $f(x,y) = (x+y)^2 + \frac{(x-y)^2}{10}$ (c) Beale loss function (d) Rosenbrock loss function

Figure 1: Comparison of optimization trajectories for various loss functions.



(a) input-neuron (b) output-neuron (c) partially group (d) filter-wise group

Figure 2: Examples of coordinate grouping. The weights with same color belong to the same group.

are based on such diagonal adaptation. Their general formulation are given in the Appendix E.

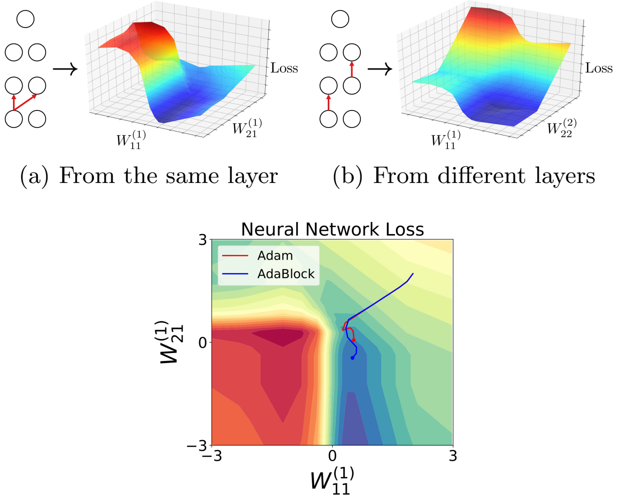
Duchi et al. (2011) also discussed the case where full-matrix adaptation can converge faster than its popular diagonal counterpart. Motivated by this, we first checked through a toy MLP experiment whether preconditioning with exact GOP (Gradient of Outer-Product, $g_t g_t^\mathsf{T}$) in (1) can be more effective even in the deep learning context. Our experiment, provided in Appendix, showed that one can achieve faster convergence and better objective values by considering the interaction between gradient coordinates (1). The caveat here is that full GOP adaptation in deep learning optimization is computationally intractable due to the square root operator in (1). Nevertheless, *is the best choice to simply use diagonal approximation given the available computation budget? What if we can afford to pay a little bit more for our computations?*

We address the above question and provide a family of adaptive SGD bridging exact GOP adaptation and its diagonal approximation, via coordinate partitioning.

Adaptive SGD with Block Diagonal Adaptation. Given a coordinate partition, we simply ignore the interactions of coordinates between different groups. For instance, given a gradient $g \in \mathbb{R}^6$, one example of constructing block diagonal matrices via coordinate partitioning is $g = (\underbrace{g_1, g_2}_{\mathcal{G}_1}, \underbrace{g_3, g_4, g_5}_{\mathcal{G}_2}, \underbrace{g_6}_{\mathcal{G}_3}) \to [g_{\mathcal{G}_1} g_{\mathcal{G}_1}^\mathsf{T} \mid 0 \mid 0 \,;\, 0 \mid g_{\mathcal{G}_2} g_{\mathcal{G}_2}^\mathsf{T} \mid 0 \,;\, 0 \mid 0 \mid g_{\mathcal{G}_3} g_{\mathcal{G}_3}^\mathsf{T}]$ where $\mathcal{G}_i$ represents each group and $g_{\mathcal{G}_i}$ denotes the collection of entries corresponding to group $\mathcal{G}_i$. Both exact GOP and diagonal approximation are special cases of our family. Exploring the use of block-diagonal matrices was suggested as future work in Duchi et al. (2011), and our work therefore provides an in-depth study of this proposal in a more generalized form. Algorithm 1 formalizes our approach for a total r groups where each group $\mathcal{G}_i$ has a size of n_i for $i \in [r]$. The Algorithm 1 can handle arbitrary grouping with appropriate reordering of entries, and groups of unequal sizes.

Note that AdaBlock allows for complete freedom to select block diagonal groups, providing a wider trade-off between computations and performance while KFAC or Shampoo use the block diagonal structure in a limited way, which incurs prohibitive memory cost for deep learning. More discussions on other quadratic or block diagonal approaches are in the Appendix.

Effect of Grouping on Optimization. Inspired by Zhuang et al. (2020), we compare the optimization trajectories for various loss functions. Here, we use the block diagonal version of Adam (called AdaBlock) and usual Adam for comparison and set the same hyperparameters. Figure 1 illustrates the trajectories. For all loss functions considered, AdaBlock shows faster convergence than Adam, and finds more accurate solution close to the optimal point. As discussed in Zhuang et al. (2020), the loss functions in Figure 1 are simple, yet they give important clues for the local behavior in deep learning optimization. Most neural



(a) From the same layer (b) From different layers

(c) Neural Network Loss

Figure 3: (a,b): Loss surfaces for different grouping methods; (c): trajectories on the loss surface (a).

networks use non-smooth ReLU activations, which the loss landscape in Figure 1-(a) reflects to some extent. Also, it is well-known that neural networks are generally ill-conditioned, and Figure 1-(a)∼(d) are similar cases.

Now, we move onto real deep learning examples. First of all, we introduce some practical grouping strategies for block diagonal adaptation. Figure 2 shows such examples in the context of deep learning models: grouping the weights with the same color in a network can approximate the exact GOP matrix with a block diagonal matrix of several small full matrices. To see which grouping could be more effective in terms of optimization, we revisit our MLP toy example. Figure 3-(a,b) show the loss landscapes for different grouping strategies (weights other than shown are fixed as true model values). We can see that the loss landscape when grouping weights in the same layer has a much more dynamic curvature than when grouping weights in different layers. In this context, we expect that a block-diagonal preconditioner is effective in terms of optimization and illustrate this empirically by comparing the grouping version for the loss landscape with dynamic curvature (Figure 3-(a)), and its diagonal counterpart. As in Figure 1, we compare both approaches using Adam. Figure 3-(c) shows the optimization histories. AdaBlock converges to a stationary point in fewer steps than usual Adam and shows a more stable trajectory.

Comparison with KFAC (Martens and Grosse, 2015). KFAC exploits a *full-matrix of curvature approximated by Kronecker product for each layer parameter*, which only allows for layer-wise block diagonal structure. AdaBlock has more freedom in constructing the block diagonal structure since the coordinate partitioning $\mathcal{P}$ in Algorithm 1 can be arbitrary. In our experiments of Section 5, we consider multiple groups *within a single layer* for efficiency, which is not possible for KFAC. Also, KFAC for conv layers is significantly slower than for fc layers; thus KFAC shows better efficiency than Sgd *only in certain cases*. In our large-scale experiments of Section 5 with many conv layers, we could not avoid an approximation where KFAC computes the inverse of curvature matrix once every 20 ∼ 100 iterations while our AdaBlock computes the inverse of curvature matrix at every iteration.

Algorithm 1 AdaBlock: **Ada**ptive Gradient Methods with **Block** Diagonal Matrix Adaptation

```
Input: Stepsize $\alpha_t$, initial point $x_1 \in \mathbb{R}^d$, and $\{\beta_{1,t}\}_{t=1}^T \in [0,1)$. The function $H_t$ designs $\widehat{V}_t \succeq 0$ with dynamic size of $r$ blocks, $\{\widehat{V}_{t,[j]}\}_{j=1}^r$.
Initialize: $m_0 = 0$, $\widehat{V}_0 = 0$.
Require: Coordinate partition $\mathcal{P}$, $\widehat{V}_t \succeq 0$.
for $t = 1, 2, \dots, T$ do
    Draw a minibatch sample $\xi_t$ from $\mathbb{P}$
    $g_t \leftarrow \nabla f(x_t)$
    $m_t \leftarrow \beta_{1,t} m_{t-1} + (1 - \beta_{1,t}) g_t$
    for $j = 1, 2, \dots, r$ do
        $\widehat{V}_{t,[j]} \leftarrow H_t(g_{1,[j]}, \cdots, g_{t,[j]};\ \mathcal{P})$
    end for
    $x_{t+1} \leftarrow x_t - \alpha_t (\widehat{V}_t^{1/2} + \delta I)^{-1} m_t$    ▷ Update rule
end for
```

3 Analysis of AdaBlock

In this section, we provide convergence and generalization analysis for AdaBlock, Algorithm 1. In addition, we study how the block size b affects our analysis.

3.1 Convergence in Non-convex Optimization

We start with the convergence of Algorithm 1. We consider the following optimization problem, $\min. f(x) := \mathbb{E}_{\xi\sim\mathbb{P}}\big[f(x;\xi)\big]$ where x is an optimization variable and ξ is a random variable representing randomly selected data sample from training data S. As in other works on non-convex optimization such as Ghadimi and Lan (2013, 2016), we study the convergence to *stationarity* and hence derive the upper bound of $\|\nabla f(x)\|^2$ by Algorithm 1, under the following mild conditions:

Assumption 1. ***(a)*** *f is differentiable, L-smooth, and lower bounded.* ***(b)*** *We assume the true gradient $\nabla f(x_t)$ and noisy gradient g_t are both bounded, i.e. $\|\nabla f(x_t)\|, \|g_t\| \le G$ for all t.* ***(c)*** *g_t is unbiased and the noise is independent, i.e. $g_t = \nabla f(x_t) + \zeta_t$ where $\mathbb{E}[\zeta_t] = 0$ and $\zeta_t \perp\!\!\!\perp \zeta_s$ for $t \neq s$.* ***(d)*** *The se-*

quence of $\{\beta_{1,t}\}_{t=1}^T$ in Algorithm 1 is non-increasing. ***(e)*** $\|\alpha_t \widehat{V}_t^{-1/2} m_t\| \leq D$ *for some strictly positive* $D > 0$.

The condition (a) is a key assumption in general nonconvex optimization analysis, and (b)-(d) are standard ones in the line of work on SGD analysis such as Chen et al. (2019). The last condition (e) states that the final step vector $\alpha_t \widehat{V}_t^{-1/2} m_t$ should be finite, which is very mild. We are ready to state our main theorem.

Theorem 1. *Let* $Q_t := \|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t \widehat{V}_t^{-1/2}\|_2 = \max_{j\in[r]}\{\|\alpha_{t-1}\widehat{V}_{t-1,[j]}^{-1/2} - \alpha_t \widehat{V}_{t,[j]}^{-1/2}\|_2\}$ *measure the maximum difference in effective spectrums over all diagonal blocks* $\widehat{V}_{t,[j]}$ *and* $\gamma_t := \lambda_{\min}(\alpha_t \widehat{V}_t^{-1/2})$. *Then, under Assumption 1, Algorithm 1 is guaranteed to yield*

$$\min_{t\in[T]} \left[\|\nabla f(x_t)\|^2\right]$$
$$\leq \frac{\mathbb{E}\left[C_1 \overbrace{\sum_{t=1}^{T} \left\|\alpha_t \widehat{V}_t^{-\frac{1}{2}} g_t\right\|^2}^{\text{Term A}} + C_2 \overbrace{\sum_{t=2}^{T} Q_t}^{\text{Term B}} + C_3 \sum_{t=2}^{T-1} Q_t^2\right] + C_4}{\displaystyle\sum_{t=1}^{T} \gamma_t} \tag{2}$$

where $\{C_i\}_{i=1}^3$ *are constants independent of problem dimension* d *and the total iterations* T, *and* C_4 *is a constant independent of* T. *We let the upper bound (2) be* $s_1(T)/s_2(T)$.

• Challenges of our analysis against prior work. The distinctly different part between the previous study (Chen et al., 2019) and AdaBlock is how to handle the term $M_t := \|(\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t \widehat{V}_t^{-1/2}) m_{t-1}\|_2$. Since $\widehat{V}_\tau$ is diagonal for the case of Chen et al. (2019), this study decomposes M_t coordinate-wisely, which is impossible in our case. Instead, we could bypass this issue using the matrix-vector inequality, $\|Ax\|_2 \leq \|A\|_2 \|x\|_2$, to bound the terms related to M_t. This trick brings *great improvements in Term B*: the Term B in Chen et al. (2019) which is the special case of (2) with $b = 1$ in our AdaBlock framework involves $\|\alpha_{t-1}/\sqrt{\widehat{v}_{t-1}} - \alpha_t/\sqrt{\widehat{v}_t}\|_1$ while ours is $\|\alpha_{t-1}/\sqrt{\widehat{v}_{t-1}} - \alpha_t/\sqrt{\widehat{v}_t}\|_\infty$.

While Theorem 1 is generally applicable to any Adam-based block-diagonal adaptation, the effect of block size b is implicitly represented in (2). To figure out the benefits of using block size $b > 1$, we study the special cases of (i) AdaGrad and (ii) EMA-based algorithms.

• Convergence of AdaGrad. We can instantiate our theorem for block diagonal extensions of AdaGrad/AdaFom (Zou and Shen, 2018; Chen et al., 2019) where the benefit of AdaBlock is explicit:

Corollary 1 (AdaGrad/AdaFom)**.** *Consider the block diagonal extension of* AdaGrad/AdaFom *with the stepsize* $\alpha_t = \frac{\alpha}{\sqrt{t}}$ *under Assumption 1 (in case of* AdaGrad, $\beta_{1,t} = 0$*). Then, they achieve* $s_1(T) = \mathcal{O}(\underbrace{\log|\widehat{V}_T| + \log T}_{\text{From Term A}} + \underbrace{1}_{\text{From Term B/others}})$ *and* $s_2(T) = \Omega(\sqrt{T})$*, hence we have* $\min_{t\in[T]} \mathbb{E}\left[\|\nabla f(x_t)\|^2\right] = \mathcal{O}(\log|\widehat{V}_T|/\sqrt{T} + \log T/\sqrt{T} + 1/\sqrt{T})$.

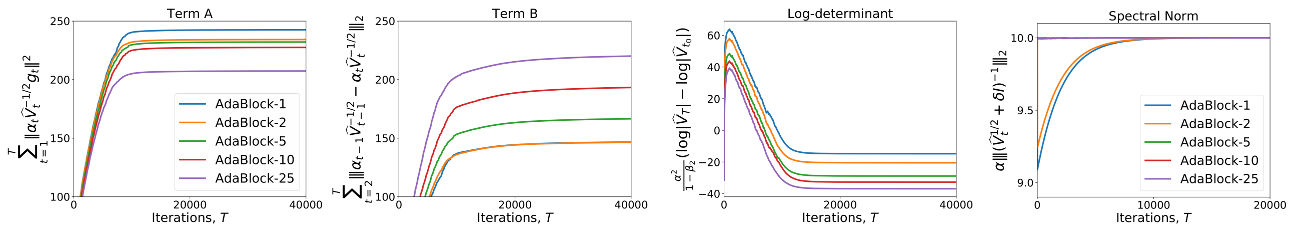
In order to see the effect of block size on convergence, we need the following lemma.

Lemma 1 (Fischer's inequality)**.** *For a positive definite matrix* $A \in \mathbb{R}^{n\times n}$, *let* $B \in \mathbb{R}^{k\times k}$ *and* $C \in \mathbb{R}^{n-k\times n-k}$ *be top left corner of* A *and bottom right corner of* A *respectively. Then,* $\det(A) \leq \det(B)\det(C)$ *holds.*

In Corollary 1, the effect of using blocks is now evident. Since Term A is asymptotically slower than Term B, Term A determines the final convergence rate and is proportional to the log-determinant of $\widehat{V}_T$. By Lemma 1, $\log|\widehat{V}_T|$ decreases as a block size b increases, so the block diagonal extension of AdaGrad theoretically achieve faster convergence than usual diagonal AdaGrad.

• Convergence of EMA-based Algorithms. Now, we consider popular EMA-based algorithms such as Adam, i.e., the design function H_t in Algorithm 1 constructs $\widehat{V}_t$ as $\widehat{V}_t = \beta_2 \widehat{V}_{t-1} + (1-\beta_2) g_t g_t^\mathsf{T}$ with $\beta_2 \in [0, 1)$. For this family, the advantage of large b for non-convex problems is not as evident as for the AdaGrad cases. Nevertheless, by Proposition 1 in Appendix E.2, Term A for EMA-based algorithms depends on $\log|\widehat{V}_T|$ similarly to the block diagonal extension of AdaGrad in Corollary 1 while Term B can be bounded by a constant regardless of block size. In that sense, we expect that Term A/Term B of EMA-based methods also have similar dynamics as those of AdaGrad in Corollary 1. For empirical studies, we design a simple experiment with MLP 784-100-10 on MNIST dataset. We optimize the parameters via block diagonal extension of Adam. The Figure 4-(a) illustrates the Term A/Term B/Logdet for $\alpha_t = 10^{-3}$. In Figure 4-(a), both Term A and $\log|\widehat{V}_T|$ decreases as a block size b increases, which corroborates Proposition 1.

• Advantages of Adaptive Gradient Methods. Importantly, we show that $\log|\widehat{V}_T|$ would affect the convergence according to Corollary 1 and empirical evidence for AdaGrad and EMA-based methods respectively. Under this intuition, the adaptive methods could achieve the smaller $\log|\widehat{V}_T|$ than vanilla Sgd in some situation. To make things clear, let us compare diagonal Adam and Sgd. The adaptation matrix $\widehat{V}_T$ for diagonal Adam is constructed as $\widehat{V}_{T,ii} = \sqrt{(1-\beta_2)\sum_{t=1}^T \beta_2^{T-t} g_{t,i}^2}$ as is known. To make



(a) Term A/Term B/Log-determinant (b) Term D

Figure 4: Empirical studies with block diagonal extension of Adam with $\alpha_t = 10^{-3}$, $\beta_{1,t} = 0.9$, and $\beta_2 = 0.999$.

$\log|\widehat{V}_T| = \sum_{i=1}^{d} \log(\widehat{V}_{T,ii})$ small, each $\widehat{V}_{T,ii}$ should be small, which means that each $g_{t,i}$ is small. This can happen *when the data feature is sparse*, which coincides with the convex regret theory for adaptive methods (Duchi et al., 2011; Kingma and Ba, 2015; Reddi et al., 2018). On the other hand, $\log|\widehat{V}_T|$ is at least zero for Sgd regardless of the size of gradient since the $\widehat{V}_T$ is just the identity matrix while $\log|\widehat{V}_T|$ could be *negative* for adaptive methods. Since we already show that using block size $b > 1$ can lead to better convergence in our paper, the benefits of AdaBlock over Sgd become evident in theory.

3.2 Uniform Stability and Generalization Error Bounds of Algorithm 1

The generalization error of a randomized algorithm A (e.g., SGD) on training data S is defined as $\epsilon_{\text{gen}} := \mathbb{E}_{S,A}\big[R_S(A(S)) - R(A(S))\big]$ where R_S and R are empirical and population risk respectively. Here, $A(S)$ means the output parameter trained with algorithm A on training data S. Hardt et al. (2015) show that an ϵ_{stab}-uniformly stable algorithm satisfies $|\epsilon_{\text{gen}}| \le \epsilon_{\text{stab}}$ where ϵ_{stab}-uniform stability is defined by

Definition 1 (Hardt et al. (2015))**.** *Let $\mathcal{D}$ be a (unknown) data distribution. For all datasets $S, S' \in \mathcal{Z}^n$ such that $|S| = |S'| = n$, and S and S' differ in only one example. The Algorithm A is said to be ϵ_{stab}-uniformly stable if $\sup_{z\in\mathcal{D}} \mathbb{E}_A\big[f(A(S); z) - f(A(S'); z)\big] \le \epsilon_{\text{stab}}$.*

In order to bound the generalization error using the result of Hardt et al. (2015), it would suffice under a Lipschitz continuity as in our Assumption 1-(b) to show that $\mathbb{E}_A\big[\|\theta - \theta'\|_2\big]$ is bounded since $\sup_{z\in\mathcal{D}} \mathbb{E}_A\big[f(A(S); z) - f(A(S'); z)\big] \le G\mathbb{E}_A\big[\|\theta - \theta'\|_2\big]$. Here, we consider an EMA-based design function H_t and defer the result of AdaGrad to Appendix.

Theorem 2 (EMA methods)**.** *Let θ_t (or θ'_t, resp.) be the trained parameter with algorithm A on training data S (or S', resp.) and $\Delta_t := \|\theta_t - \theta'_t\|_2$. Further, let t_0 denote the time when $\widehat{V}_t$ and $\widehat{V}'_t$ becomes full-rank. For $\alpha_t = \alpha$ and $\beta_{1,t} = 0$, we have the recurrence relation,*

$$\mathbb{E}\big[\Delta_{T+1}\big] \le \frac{\alpha\sqrt{T}}{n\sqrt{1-\beta_2}}\Big[\sqrt{g(\widehat{V}_T)} + \sqrt{g(\widehat{V}'_T)}\Big] + \alpha\Big(1 - \frac{1}{n}\Big)J_T$$

with the quantities

$$g(\widehat{V}_T) = \frac{t_0(1-\beta_2)G^2}{\delta^2} + \mathbb{E}\big[\overbrace{\log\frac{|\widehat{V}_T|}{|\widehat{V}_{t_0}|}}^{\text{Term C}}\big] + d(T - t_0)\log\frac{1}{\beta_2}$$

$$J_T = G\sum_{t=1}^{T}\mathbb{E}\Big[\underbrace{\vert\!\vert\!\vert(\widehat{V}_t^{1/2} + \delta I)^{-1}\vert\!\vert\!\vert_2}_{\text{Term D}} + \underbrace{\vert\!\vert\!\vert(\widehat{V}_t'^{1/2} + \delta I)^{-1}\vert\!\vert\!\vert_2}_{\text{Term D}'}\Big] + L\sum_{t=1}^{T}\mathbb{E}\Big[\underbrace{\vert\!\vert\!\vert(\widehat{V}_t'^{1/2} + \delta I)^{-1}\vert\!\vert\!\vert_2}_{\text{Term D}'}\Delta_t\Big]$$

Theorem 2 describes ϵ_{stab} for AdaBlock, hence we analyse how small the upper bound for $\mathbb{E}[\Delta_{T+1}]$ is according to block sizes. In the quantities $g(\widehat{V}_T)$ and J_T, we remark the Term C/Term D/Term D′ since only they depend on the block sizes. Therefore, we investigate the dynamics of Term C/Term D/Term D′.

- **Dynamics of Term C/Term D/Term D′.** Since Term D and Term D′ have exactly same dynamics, we only discuss Term D. In Theorem 2, the Term C is smallest when $b = d$ as in Corollary 1. Term D is smallest for $b = 1$ since $\max_i A_{ii} \le \lambda_{\max}(A)$ for any matrix $A \in \mathcal{S}_{++}$. By the way, the Term D can be bounded as $\vert\!\vert\!\vert(\widehat{V}_t^{1/2} + \delta I)^{-1}\vert\!\vert\!\vert_2 \le 1/\delta$ which is independent of T. For empirical studies, we revisit our experiment with MLP 784-100-10 on MNIST dataset. The Figure 4-(b) shows that $\alpha\vert\!\vert\!\vert(\widehat{V}_t^{1/2} + \delta I)^{-1}\vert\!\vert\!\vert_2$ converges to $\frac{\alpha}{\delta}$ (Here, we set $\frac{\alpha}{\delta} = \frac{10^{-3}}{10^{-4}} = 10$) regardless of block sizes and the difference among block sizes is

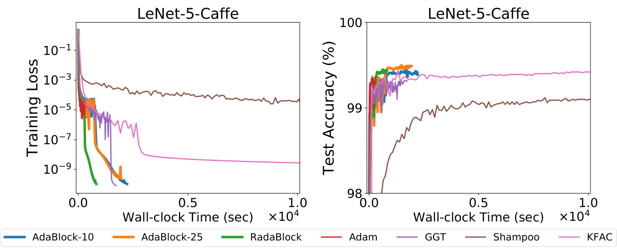


Figure 5: Results on LeNet-5 on MNIST.

negligible compared to $\log|\widehat{V}_T|$ (see Figure 4). As a result, Term C is dominant in terms of block size, so we can expect that the upper bound for $\mathbb{E}[\Delta_T]$ is smaller for $b > 1$ than $b = 1$.

- **Large δ improves generalization.** Zaheer et al. (2018) suggest using large δ to improve generalization but with only empirical studies. In Theorem 2, it can be seen clearly that the upper bound for $\mathbb{E}[\Delta_T]$ is smaller for large δ, which in result improves generalization.

The analysis on time complexity and memory footprint against various baselines is provided in the Appendix.

4 Practical Extensions of AdaBlock

4.1 Spectrum-Clipping for Improving Generalization

Wilson et al. (2017) showed that adaptive methods are better than vanilla SGD in the early stage but get worse as training matures. To address this, Keskar and Socher (2017) suggests training networks with ADAM at the beginning and then switching to SGD. Luo et al. (2019) proposes ADABOUND which clips the effective learning rate $\alpha_t/(\sqrt{v_t}+\epsilon)$ of ADAM by decreasing sequence of intervals $I_t = [\eta_l(t), \eta_u(t)]$ at every iteration which converges to some point, thereby resembling SGD in the end. However, such an extension is not obvious for ADABLOCK due to the absence of *effective* stepsize. Instead, we propose a *spectrum-clipping* which clips the spectrum of $\alpha_t(\widehat{V}_t^{1/2}+\delta I)^{-1}$ by decreasing sequence of intervals. We use the modified update rule in Algorithm 1 after constructing $\widehat{V}_t$: **(i)** $\widehat{U}_t, \widehat{\Sigma}_t^{1/2}, \_ \leftarrow \mathrm{SVD}(\widehat{V}_t^{1/2})$, **(ii)** $\widetilde{\Sigma}_t^{-1/2} \leftarrow \mathrm{Clip}(\lambda(\alpha_t(\widehat{\Sigma}_t^{1/2}+\delta I)^{-1}), \lambda_l(t), \lambda_u(t))$, and **(iii)** $x_{t+1} \leftarrow x_t - \widehat{U}_t^{\top}\widetilde{\Sigma}_t^{-1/2}\widehat{U}_t m_t$. We schedule the clipping intervals converging to a single point uniformly over the spectrum so that $\alpha_t(\widehat{V}_t^{1/2}+\delta I)^{-1}$ can be easily computed in the form of constant times identity matrix and effectively behaves like vanilla SGD.

Table 1: Test accuracy for LeNet-5-Caffe experiment.

Algorithm	Accuracy (%)
ADAM (Kingma and Ba, 2015)	99.38 ± 0.132
TONGA (Le Roux et al., 2007)	99.27 ± 0.147
KFAC (Martens and Grosse, 2015)	99.44 ± 0.103
SHAMPOO (Gupta et al., 2018)	99.18 ± 0.173
GGT (Agarwal et al., 2019)	99.35 ± 0.082
ADABLOCK-10	**99.44** ± 0.096
ADABLOCK-25	**99.51** ± 0.112
RADABLOCK	**99.42** ± 0.078

4.2 RAdaBlock: Randomized Layer-wise Block Diagonal Adaptation

To further reduce computational cost and memory footprint for practical purpose, we propose the following randomized update. At each iteration, one selects ℓ layers at random to be updated via block diagonal adaptation as in vanilla ADABLOCK, while the remaining layers are updated via the usual diagonal approach. Our experiments in Section 5 will confirm that such a scheme can combine the advantages of block-diagonal and diagonal adaptation: fast per-iteration time, small memory footprint and better generalization.

5 Experiments

In this section, we design three sets of experiments, each of which considers different variant of ADABLOCK and corresponding baselines depending on the purpose of experiments. The first set considers vanilla ADABLOCK to verify the effect only from coordinate grouping. The second set investigates whether block diagonal matrix adaptation with spectrum-clipping can achieve state-of-the-art performance. The third set evaluates RADABLOCK, the randomized version of ADABLOCK.

In our algorithms, coordinate grouping can be done in several ways. Given our insight that grouping weights in the same layer could be more effective, we employ Figure 2-(c) grouping $10 \sim 32$ parameters connected to input-neuron for fc layer and filter-wise grouping for conv layers in Figure 2-(d). In all experiments, we use the block diagonal versions of ADAM as representative of ADABLOCK since ADAM is the most frequently used in deep learning. Note that the block diagonal extension of ADAGRAD (Duchi et al., 2011), RMSPROP (Tieleman and Hinton, 2012), ADABELIEF, and many other optimizers could be naturally considered.

Details on experimental settings and additional considerations on computations/memory are provided in Appendix.

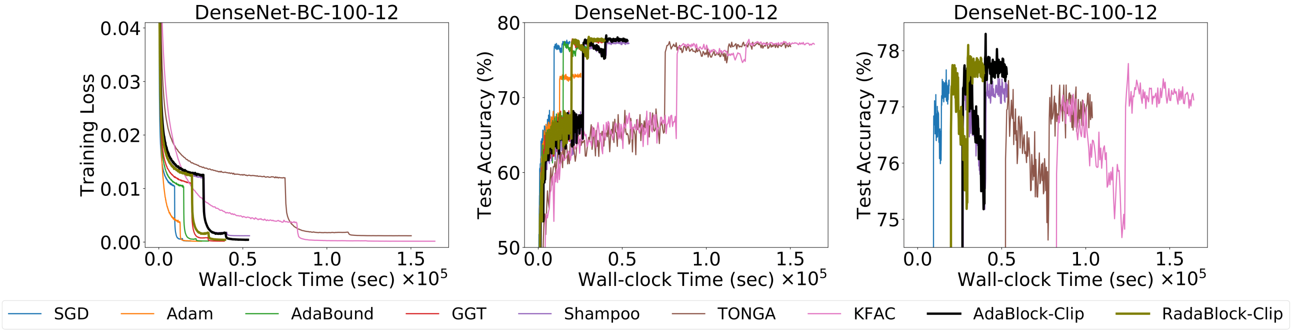


Figure 6: Results on spectrum-clipping for DenseNet on CIFAR-100 dataset.

Table 2: Test accuracy for DenseNet experiment.

Algorithm	Accuracy (%)
SGD + Momentum	77.81 ± 0.137
ADAM	73.53 ± 0.121
ADABOUND	77.62 ± 0.156
KFAC	78.01 ± 0.077
SHAMPOO	77.74 ± 0.149
GGT	77.90 ± 0.114
TONGA	77.40 ± 0.093
ADABLOCK-CLIP	**78.25** ± 0.102
RADABLOCK-CLIP	**78.15** ± 0.080

5.1 Investigating Grouping Effect

We investigate the effect of coordinate grouping on **(i)** MNIST classification and **(ii)** Variational autoencoder. The latter is presented in the Appendix.

MNIST Classification. We consider a simple LeNet-5 network. We use 128 mini-batch size and train networks with 100 epochs. For fair comparison, we compute the inverse of preconditioners *every iteration* for KFAC, Shampoo, and ADABLOCK. As Figure 5 illustrates the results, the learning curve looks similar in the early stage of training, but ADABLOCK converges without oscillations in the latter part of training, which corroborates the effect of block sizes in Theorem 1. Importantly, ADABLOCK achieves faster convergence and lower objective values than any other previous methods approximating the full-matrix preconditioners such as KFAC/Shampoo/GGT, which corroborates asymptotic wall-clock time comparisons in Appendix B. The generalization of ADABLOCK becomes more stable than diagonal variant and GGT, and overall superior across epochs, which can be seen in Table 1.

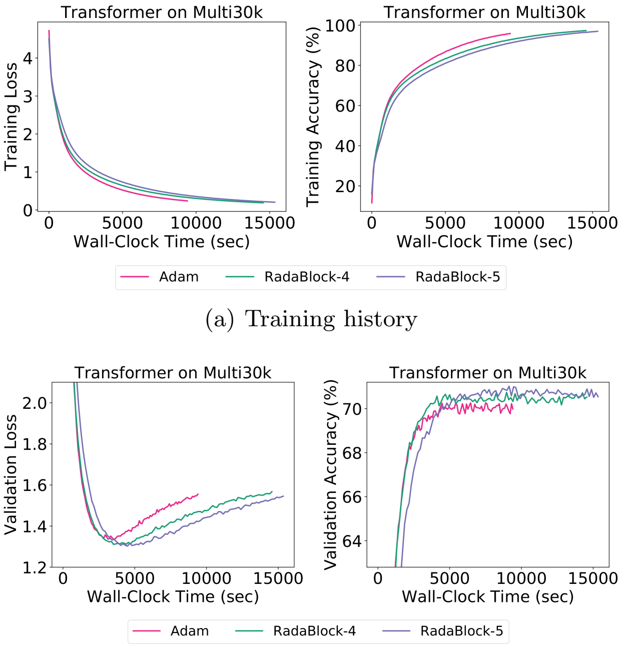
5.2 Evaluating Spectrum-Clipping

We demonstrate our algorithms using more complex benchmark architecture/dataset for image classification. For this task, vanilla SGD with proper learning rate scheduling has enjoyed state-of-the-art performance. Therefore, we compare algorithms using our spectrum-clipping methods that can exploit higher generalization ability of vanilla SGD. Especially for Shampoo and KFAC, we compute the inverse of preconditioners every 20 iteration for practical purpose, but for our ADABLOCK we calculate the inverse of preconditioner *every iteration*. We train DenseNet (Huang et al., 2017) on CIFAR-100 dataset and Figure 6 illustrates our results. The training speed of ADABLOCK is a little slower but practically acceptable. Notably, ADABLOCK achieves the best generalization performance among all comparison optimizers including KFAC, Shampoo, and GGT. Specifically, the improvement in performance is about 0.5% over vanilla SGD as can be seen in Table 2. We report the actual memory usage among modified full-matrix adaptations in Table 5 in Appendix B.3.

5.3 Evaluating RadaBlock

We now evaluate the randomized version RADABLOCK. Considering various problem domains and diverse large-sized network architectures, we consider (i) GANs, (ii) Transformer, (iii) ImageNet classification. Note that the computational cost of ADABLOCK depends heavily on the network structure rather than the dataset. The results on GANs are provided in the Appendix.

Neural Machine Translation. Since Transformer-based models (Vaswani et al., 2017) become standard for natural language understanding tasks, we consider the neural machine translation with Transformer on Multi30k dataset (Elliott et al., 2016). We choose 4 or 5 layers to be updated via block diagonal adaptations for RADABLOCK in order to balance the wall-clock time and performance. Figure 7 demonstrate the results.



(a) Training history

(b) Validation history

Figure 7: Results on Transformer on Multi30k dataset.

Table 3: Test BLEU for Transformer on Multi30k dataset. Adam works much better than vanilla SGD, so we compare RadaBlock with Adam. Here, KFAC/Shampoo/GGT are **excluded** due to prohibitive memory requirement.

	Test BLEU score
Adam	36.09 ± 0.135
RadaBlock-4	**36.37** ± 0.141
RadaBlock-5	**36.12** ± 0.117
Vanilla AdaBlock (N/A)	**36.71**

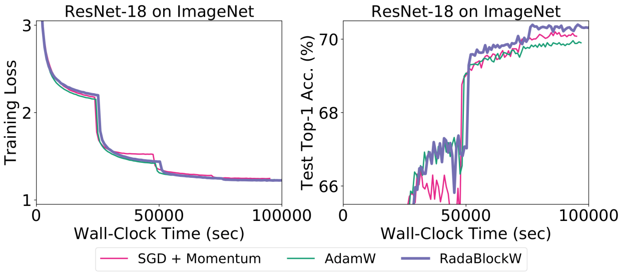


Figure 8: Results on ResNet-18 on ImageNet.

Table 4: Top-1 accuracy for training ResNet-18 on ImageNet dataset.

	Top-1 Accuracy (%)
SGDW + Momentum	70.13 ± 0.072
AdamW	70.02 ± 0.095
RadaBlockW	**70.30** ± 0.095
Vanilla AdaBlock (N/A)	**78.11**

Although the training curves show similar dynamics, RadaBlock outperforms Adam in terms of validation. Also, we can see in Table 3 that RadaBlock is superior to Adam in terms of test BLEU. In this experiment, we focus on evaluating RadaBlock but also include the results of vanilla AdaBlock in Table 3.

ImageNet Classification. Recently, Loshchilov and Hutter (2019) proposed to fix the weight decay regularization for adaptive gradient methods to achieve competitive generalization with vanilla Sgd. In this context, we consider a decoupled weight decay variant of our randomized approach, which we term RadaBlockW. We train ResNet-18 He et al. (2016) on ImageNet dataset Russakovsky et al. (2015). For RadaBlock, we randomly choose two layers to be updated via block diagonal adaptation at every iteration. We focus on evluating RadaBlock since the network/dataset are too large to evaluate vanilla AdaBlock. To clearly see the effect of our approach, we use the same weight decay value for AdamW and RadaBlockW. Figure 8 illustrates our results. The learning curves for each method show similar convergence speed, but RadaBlockW is superior to other methods in terms of generalization. We can see the generalization comparison in Table 4. Notably, AdamW and RadaBlockW differ only in update rules, but RadaBlockW outperforms AdamW. As in Transformer, we include the results of vanilla AdaBlock in Table 4 and the learning curves between vanilla AdaBlock and RadaBlock are in the Appendix.

6 Conclusion

We presented AdaBlock, a family of adaptive gradient methods that approximates exact GOP with block diagonal matrices to effectively utilize structural characteristics of deep learning architectures. Vanilla AdaBlock employs block-diagonal adaptation to the whole network, while its randomized variant RadaBlock combines block diagonal and diagonal adaptation to further reduce computational resources. We analyzed convergence and generalization of vanilla AdaBlock, highlighting benefits compared to popular diagonal counterparts. We proposed a spectrum-clipping scheme to boost generalization. Extensive experiments on various deep leaning tasks demonstrated the value of the AdaBlock framework. As future work, we plan to explore strategies for actively selecting layers benefiting most from block-diagonal adaptation at each iteration of RadaBlock, and to analyze RadaBlock in theory.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grants (No.2018R1A5A1059921, No.2019R1C1C1009192), Institute of Information & Communications Technology Planning & Evaluation (IITP) grants (No.2017-0-01779, XAI, 2019-0-00075, Artificial Intelligence Graduate School Program(KAIST) No.2019-0-01371, Development of brain-inspired AI with human-like intelligence) funded by the Korea government (MSIT).

References

Agarwal, N., Bullins, B., Chen, X., Hazan, E., Singh, K., Zhang, C., and Zhang, Y. (2019). Efficient full-matrix adaptive regularization. In *International Conference on Machine Learning*, pages 102–110.

Chen, X., Liu, S., Sun, R., and Hong, M. (2019). On the convergence of a class of adam-type algorithms for non-convex optimization. In *International Conference on Learning Representation (ICLR).*

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

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

Elliott, D., Frank, S., Sima'an, K., and Specia, L. (2016). Multi30K: Multilingual English-German image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74.

Ghadimi, S. and Lan, G. (2013). Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368.

Ghadimi, S. and Lan, G. (2016). Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1-2):59–99.

Gupta, V., Koren, T., and Singer, Y. (2018). Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine learning (ICML).*

Hardt, M., Recht, B., and Singer, Y. (2015). Train faster, generalize better: Stability of stochastic gradient descent. *arXiv preprint arXiv:1509.01240.*

Hazan, E., Agarwal, A., and Kale, S. (2007). Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192.

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*, pages 770–778.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Keskar, N. S. and Socher, R. (2017). Improving generalization performance by switching from adam to sgd. *arXiv preprint arXiv:1712.07628.*

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representation (ICLR).*

Le Roux, N., Manzagol, P.-A., and Bengio, Y. (2007). Topmoumoute online natural gradient algorithm. In *NIPS*, pages 849–856. Citeseer.

Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR).*

Luo, L., Xiong, Y., and Liu, Y. (2019). Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations (ICLR).*

Martens, J. and Grosse, R. (2015). Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417.

McMahan, H. B. and Streeter, M. (2010). Adaptive bound optimization for online convex optimization. In *Conference on Computational Learning Theory (COLT).*

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Reddi, S. J., Kale, S., and Kumar, S. (2018). On the convergence of adam and beyond. In *International Conference on Learning Representation (ICLR).*

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

Tieleman, T. and Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wang, M., Yu, L., Zheng, D., Gan, Q., Gai, Y., Ye, Z., Li, M., Zhou, J., Huang, Q., Ma, C., Huang, Z., Guo, Q., Zhang, H., Lin, H., Zhao, J., Li, J., Smola, A. J., and Zhang, Z. (2019). Deep graph library: Towards efficient and scalable deep learning on graphs. *ICLR Workshop on Representation Learning on Graphs and Manifolds.*

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., and Recht, B. (2017). The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems (NIPS).*

Yu, A. W., Huang, L., Lin, Q., Salakhutdinov, R., and Carbonell, J. (2017). Block-normalized gradient method: An empirical study for training deep neural network. *arXiv preprint arXiv:1707.04822.*

Zaheer, M., Reddi, S., Sachan, D., Kale, S., and Kumar, S. (2018). Adaptive methods for nonconvex optimization. In *Advances in Neural Information Processing Systems*, pages 9793–9803.

Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701.*

Zheng, S. and Kwok, J. T. (2019). Blockwise adaptivity: Faster training and better generalization in deep learning. *arXiv preprint arXiv:1905.09899.*

Zhuang, J., Tang, T., Ding, Y., Tatikonda, S. C., Dvornek, N., Papademetris, X., and Duncan, J. (2020). Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *Advances in Neural Information Processing Systems*, 33.

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

Supplementary Materials

A Toy MLP example: Full GOP Adaptation vs. Diagonal Approximation for Section 2

We consider a structured MLP (two nodes in two hidden layers followed by single output). For hidden units, we use ReLU activation (Nair and Hinton, 2010) and the sigmoid unit for the binary output. We generate $n = 10$ i.i.d. observations: $x_i \sim \mathcal{N}(0, I_2)$ and y_i from this two layered MLP given x_i. The results of our toy experiment are depicted in Figure 9.

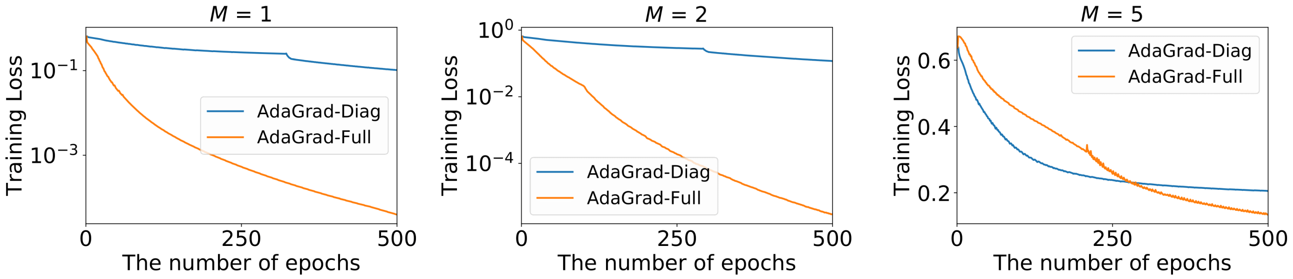


Figure 9: Comparison of AdaGrad diagonal version and full matrix version varying the minibatch size M.

B Comparison with Baselines with Computations and Memory Considerations

B.1 The Difference of KFAC/Shampoo and AdaBlock in terms of Block Diagonal Structure

B.2 Computations

Compared with full matrix adaptation, working with a block diagonal matrix is computationally more efficient as it allows for decoupling computations with respect to each small full sub-matrix. In vanilla AdaBlock (Algorithm 1), the procedures for constructing the block diagonal matrix and for updating parameters for each block by computing the "inverse" square root of each sub-matrix can be done in a parallel manner. As the group size increases, the block diagonal matrix becomes closer to the full matrix, resulting in greater computational cost. Therefore, it makes sense to consider relatively small group size in practice. We consider the single layer with the weight parameter of size $m \times m$ for easy comparison. The main bottleneck in terms of time complexity is computing the inverse operations. For such layer, AdaBlock requires batch mode of SVD operation to compute the inverse of preconditioner, which takes $\mathcal{O}(b^3 \times \frac{m^2}{b}) = \mathcal{O}(b^2 m^2)$ for block size b. On the other hand, both Shampoo and KFAC utilize Kronecker product where each Kronecker factor has size of $m \times m$, so they require $\mathcal{O}(m^3 + m^3) = \mathcal{O}(2m^3)$ for matrix inverse operations to compute the inverse of preconditioners (since the inverse of Kronecker product is the product of inverse Kronecker factors). Hence, AdaBlock is very advantageous in terms of time complexity as long as roughly $b \leq \sqrt{2m}$ is satisfied. We consider small block size and large hidden units, so the condition $b \leq \sqrt{2m}$ almost always holds in practice. For such reasons,AdaBlock can compute the inverse of preconditioner *every iteration*, but Shampoo/KFAC compute it every $20 \sim 100$ iteration inevitably in order to reduce heavy computational overhead.

The wall-clock time performance of AdaBlock can be further improved using RAdaBlock. As can be seen in Figure **??**-(a), RAdaBlock achieves higher inception score in wall-clock time.

B.3 Memory

In terms of memory, vanilla AdaBlock is more efficient than GGT Agarwal et al. (2019). For example, consider models with a total of d parameters. For Algorithm 1, assume that $\widehat{V}_t$ is a block diagonal matrix with r sub-matrices, and each block has size $b \times b$ (so, $br = d$). Also, assume that the truncated window size for GGT is w. GGT needs a memory size of $\mathcal{O}(wd)$, and our algorithm requires $\mathcal{O}(rb^2) = \mathcal{O}(bd)$. We consider small group size $b = 10$ or 25 for our experiments while the recommended window size of GGT is 200 Agarwal

Table 5: GPU memory consumptions among modified full-matrix adaptation methods for DenseNet on CIFAR-100 experiments in Section 5.2. We use machine with Intel(R) Xeon(R) CPU E5-2630v4 @2.20GHz and Titan Xp GPUs.

KFAC	Shampoo	GGT	AdaBlock-Clip
(Martens and Grosse, 2015)	(Gupta et al., 2018)	(Agarwal et al., 2019)	(Ours)
3261MiB	3033MiB	3617MiB	3029MiB

et al. (2019). Therefore, our algorithm is more memory-efficient and the benefit is more pronounced as the number of model parameters d is large, which is the case in popular deep learning models/architectures. For other modified full-matrix adaptations, we consider the single layer with the weight parameter of size $m \times m$ as in comparison of computations. Since AdaBlock should save $b \times b$ blocks for all batches, AdaBlock requires $\mathcal{O}(\frac{m^2}{b} \times b \times b) = \mathcal{O}(bm^2)$ in memory. On the other hand, both Shampoo and KFAC should save each Kronecker factor, which requires $\mathcal{O}(m^2 + m^2) = \mathcal{O}(2m^2)$. However, the matrix inversion requires $\mathcal{O}(b^3)$ and $\mathcal{O}(m^3)$ memory for AdaBlock and KFAC respectively, so the total memory requirement would be finally $\mathcal{O}(bm^2 + \frac{m^2}{b}b^3) = \mathcal{O}(b^2m^2 + bm^2) = \mathcal{O}(b^2m^2)$ and $\mathcal{O}(2m^3 + 2m^2) = \mathcal{O}(2m^3)$ respectively. Since the number of neurons m is very large in general for modern network architecture and we choose a small block size such that $b \ll \sqrt{2m}$, AdaBlock requires much less memory than KFAC. We include the exact memory consumptions in Table for DenseNet experiments on CIFAR-100. As seen in Table 5, the memory usage for AdaBlock is not much compared to KFAC/Shampoo/GGT, but rather our AdaBlock uses the least memory.

C Hyperparameters and Additional Experimental Results for Section 5

We use the recommended step size or tune it in the range $[10^{-4}, 10^{2}]$ for all comparison algorithms. For Adam based algorithms, we use default decay parameters $(\beta_1, \beta_2) = (0.9, 0.999)$. For a diagonal version of Adam variant algorithm, we choose numerical stability parameter $\epsilon = 10^{-3}$ since the larger value of ϵ can improve the generalization performance as discussed in Zaheer et al. (2018). For spectrum-clipping in Section 4.1, we use the same intervals $\lambda_l(t) = \alpha^*\Big(1 - \dfrac{1}{\gamma t + 1}\Big)$ and $\lambda_u(t) = \alpha^*\Big(1 + \dfrac{1}{\gamma t}\Big)$ as in Luo et al. (2019). For γ and α^* in clipping bound functions, we consider $\gamma \in \{0.0001, 0.0005, 0.001\}$ and choose $\alpha^* \in \{\alpha_{\text{SGD}}, 5\alpha_{\text{SGD}}, 10\alpha_{\text{SGD}}\}$ where α_{SGD} is the best-performing initial learning rate for vanilla Sgd (These hyperparameter candidates are based on the empirical studies in Luo et al. (2019)). As in Luo et al. (2019), our results are also not sensitive to choice of γ and α^*. With these hyperparameters, we consider maximum 300 epochs training time, and mini-batch size or learning rate scheduling are introduced in each experiment description. Our Algorithm 2 requires SVD procedures to compute the square root of a block diagonal matrix. We apply SVD efficiently to all small sub-matrices simultaneously through batch mode of SVD. Especially for KFAC, we use fixed damping parameter 10^{-3} and tune the learning rate.

MNIST Classification. We consider the following LeNet-5 network architecture, 20C5 - MP2 - 50C5 - MP2 - 500FC - softmax. Designing for corroborating our theoretical results, we employ the numerical stability parameter $\delta = \epsilon = 10^{-4}$.

CIFAR classification. According to experiment settings in (Huang et al., 2017), we use mini-batch size 64 and consider maximum 300 epochs. Also, we use a *step-decay* learning rate scheduling in which the learning rate is divided by 10 at 50% and 75% of the total number of training epochs. With this setting, vanilla Sgd with a momentum factor 0.9 performs best with initial learning rate $\alpha^* = 0.1$, so we use this value for our bound functions of spectrum-clipping, $\lambda_l(t)$ and $\lambda_u(t)$. As recommended in Zaheer et al. (2018), we employ $\delta = \epsilon = 10^{-3}$ for better generalization.

Neural Machine Translation. Following Vaswani et al. (2017), we employ the same learning rate annealing using Adam optimizer with decaying parameters $(\beta_1, \beta_2) = (0.9, 0.98)$. Under this setting, the initial learning rates for both Adam and RadaBlock are the same and we use the batch size 128. For this experiment, we apply our optimization algorithms upon the open source library DGL (Wang et al., 2019).

ImageNet Classification. We use the official PyTorch model repository and example codes for ResNet-18 and running script respectively. As the batch size is increased to 1024, we set the initial learning rate for each optimizer larger. Specifically, we use $\alpha = 0.2$ for vanilla SGD with momentum parameter 0.9 and $\alpha = 0.002$ for both ADAM and RADABLOCK. To see the only the effect of the block diagonal approximation, the weight decay factors for both ADAM and RADABLOCK are set equal to 0.01.

C.1 Vanilla AdaBlock vs. RadaBlock for Transformer

Here, we include the comparisons between vanilla ADABLOCK and RADABLOCK only for better understanding our approach. Although vanilla ADABLOCK takes too long per-iteration time, Figure 10 indirectly shows that the more layers are selected, the better the generalization can be.

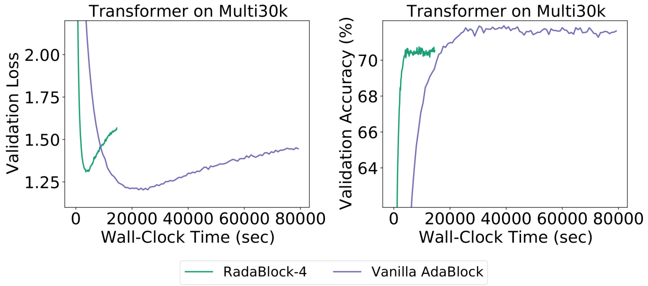


Figure 10: Validation history for Transformer.

C.2 Vanilla AdaBlock vs. RadaBlock for ImageNet Classification

As in Transformer experiment, we include the comparisons between vanilla ADABLOCK and RADABLOCK only for better understanding our approach. Although vanilla ADABLOCK takes too long per-iteration time, Figure 11 indirectly shows that the more layers are selected, the better the generalization can be.

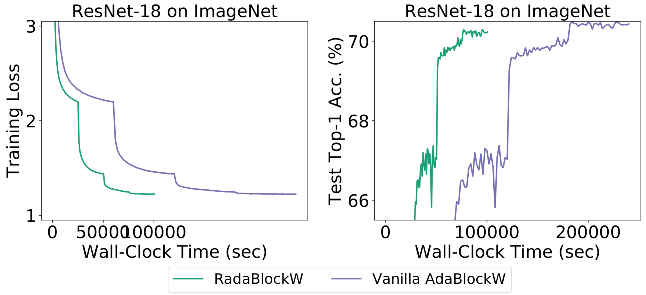


Figure 11: Training/Vaildation history for ImageNet classification.

D Detail Algorithm of AdaBlock and General Frameworks

Algorithm 2 Adaptive Gradient Methods with Block Diagonal Matrix Adaptations via Grouping (More detailed version)

```
Input: Stepsize $\alpha_t$, initial point $x_1 \in \mathbb{R}^d$, $\beta_1 \in [0, 1)$, and the function $H_t$ which designs $\widehat{V}_t$.
Initialize: $m_0 = 0$, $\widehat{V}_0 = 0$, and $t = 0$.
Assumption: We have $r$ blocks with each size $n_i \times n_i$ and $n_1 + \cdots + n_r = d$, and $\beta_{1,t} \geq \beta_{1,t+1}$
for $t = 1, 2, \ldots, T$ do
    Draw a minibatch sample $\xi_t$ from $\mathbb{P}$
    offset $\leftarrow 0$
    $G_t \leftarrow 0$
    $g_t \leftarrow \nabla f(x_t)$
    $m_t \leftarrow \beta_{1,t} m_{t-1} + (1 - \beta_{1,t}) g_t$
    for each group index $j = 1, 2, \ldots, r$ do
        $g_t^{(j)} \leftarrow g_t[\text{offset} : \text{offset} + n_j]$
        $G_t[\text{offset} : \text{offset} + n_j, \text{offset} : \text{offset} + n_j] \leftarrow g_t^{(j)} \big(g_t^{(j)}\big)^T$
        offset $\leftarrow$ offset $+ n_j$
    end for
    $\widehat{V}_t \leftarrow H_t(G_1, \cdots, G_t)$
    $x_{t+1} \leftarrow x_t - \alpha_t (\widehat{V}_t^{1/2} + \delta I)^{-1} m_t$
end for
```

Algorithm 3 General Adaptive Gradient Methods approximating $g_t g_t^T$ via DIAGONAL Matrix

```
Input: Initial point $x_1 \in \mathbb{R}^d$, stepsize $\{\alpha_t\}_{t=1}^T$, decay parameters $\beta_{1,t}, \beta_2 \in [0, 1]$, and $\epsilon > 0$.
Initialize: $m_0 = 0$, $\widehat{v}_0 = 0$.
for $t = 1, 2, \ldots, T$ do
    Draw a minibatch sample $\xi_t$ from $\mathbb{P}$
    $g_t \leftarrow \nabla f(x_t; \xi_t)$
    $G_t \leftarrow \text{diag}(g_t g_t^T)$
    $m_t \leftarrow \beta_{1,t} m_{t-1} + (1 - \beta_{1,t}) g_t$
    $\widehat{v}_t \leftarrow h_t(G_1, G_2, \ldots, G_t)$
    $x_{t+1} \leftarrow x_t - \alpha_t m_t / (\sqrt{\widehat{v}_t} + \epsilon)$
end for
Output: $\widehat{x}$.
```

Algorithm 4 General Adaptive Gradient Methods with the exact $g_t g_t^T$ (FULL Matrix)

```
Input: Initial point $x_1 \in \mathbb{R}^d$, stepsize $\{\alpha_t\}_{t=1}^T$, decay parameters $\beta_{1,t}, \beta_2 \in [0, 1]$, and $\delta > 0$.
Initialize: $m_0 = 0$, $\widehat{V}_0 = 0$.
for $t = 1, 2, \ldots, T$ do
    Draw a minibatch sample $\xi_t$ from $\mathbb{P}$
    $g_t \leftarrow \nabla f(x_t; \xi_t)$
    $G_t \leftarrow g_t g_t^T$
    $m_t \leftarrow \beta_{1,t} m_{t-1} + (1 - \beta_{1,t}) g_t$
    $\widehat{V}_t \leftarrow H_t(G_1, G_2, \ldots, G_t)$
    $x_{t+1} \leftarrow x_t - \alpha_t (\widehat{V}_t^{1/2} + \delta I)^{-1} m_t$
end for
Output: $\widehat{x}$.
```

We provide the general frameworks of adaptive gradient methods with exact full matrix adaptations. The Algorithm 3 and 4 represent the general framework for each case. We can identify algorithms according to the functions h_t (Table 6) and H_t (Table 7) which determine the dynamics of $\widehat{v}_t$ and $\widehat{V}_t$ respectively. Also, the Algorithm 2 is a detail version of the Algorithm 1.

E Details for Convergence Analysis in Section 3.1

E.1 Proofs for Corollary 1

Before moving onto proofs, we need the following technical lemma

Lemma 2 (Lemma 12 in Hazan et al. (2007)). *For positive definite matrices A and B, the following inequality holds*

$$\text{Tr}\big(A^{-1}(A - B)\big) \leq \log |A| - \log |B|$$

Table 6: Variants of diagonal matrix adaptations

$\widehat{v}_t$ \ $\beta_{1,t}$	$\beta_{1,t}=0$	$\beta_{1,t}=\beta_1$
1	SGD	-
$(1/t)\sum_{t=1}^{T} g_t^2$	ADAGRAD	ADAFOM
$\beta_2\widehat{v}_{t-1}+(1-\beta_2)g_t^2$	RMSPROP	ADAM
$v_t=\beta_2 v_{t-1}+(1-\beta_2)g_t^2$, $\widehat{v}_t=\max\{\widehat{v}_{t-1},v_t\}$	-	AMSGRAD

Table 7: Variants of full matrix adaptations

$\widehat{V}_t$ \ $\beta_{1,t}$	$\beta_{1,t}=0$	$\beta_{1,t}=\beta_1$
$\widehat{V}_t=I$	SGD	-
$\widehat{V}_t=\frac{1}{T}\sum_{t=1}^{T} g_t g_t^T$	ADAGRAD	ADAFOM
$\widehat{V}_t=\beta_2\widehat{V}_{t-1}+(1-\beta_2)g_t g_t^T$	RMSPROP	ADAM
$V_t=U_t\Sigma_t U_t^T$, $\widehat{V}_t=U_t\max\{\widehat{\Sigma}_{t-1},\Sigma_t\}U_t^T$	-	AMSGRAD

For ADAGRAD, we have $\alpha_t=\alpha/\sqrt{t}$ and $\widehat{V}_t=\frac{1}{t}\sum_{i=1}^{t} g_i g_i^T \coloneqq \frac{1}{t}\widehat{G}_t$. First, we bound the Term A/Term B to find the $s_1(T)$.

The Term A is

$$\sum_{t=1}^{T}\|\alpha_t\widehat{V}_t^{-1/2}g_t\|^2=\sum_{t=1}^{t_0}\|\alpha_t\widehat{V}_t^{-1/2}g_t\|^2+\underbrace{\sum_{t=t_0+1}^{T}\|\alpha_t\widehat{V}_t^{-1/2}g_t\|^2}_{T_1}$$

Since the first term in RHS is independent of T, so we only need to bound T_1. The quantity T_1 can be bound as

$$\begin{aligned}
T_1 &= \sum_{t=t_0+1}^{T}\|\alpha_t\widehat{V}_t^{-1/2}g_t\|^2\\
&= \alpha^2\sum_{t=t_0+1}^{T}\frac{1}{t}\|V_t^{-1/2}g_t\|^2\\
&= \alpha^2\sum_{t=t_0+1}^{T}\mathrm{Tr}\big(\widehat{V}_t^{-1}\frac{1}{t}g_t g_t^T\big)\\
&\le \alpha^2\sum_{t=t_0+1}^{T}\Big[\log\big|\widehat{V}_t\big|-\log\big|\frac{t-1}{t}\widehat{V}_{t-1}\big|\Big]\\
&= \alpha^2\sum_{t=t_0+1}^{T}\Big[\log\big|\widehat{V}_t\big|-\log\big|\widehat{V}_{t-1}\big|+d\log\frac{t}{t-1}\Big]\\
&= \alpha^2\big(\log|\widehat{V}_T|-\log|\widehat{V}_{t_0}|+d\log\frac{T}{t_0}\big)=\mathcal{O}(\log|\widehat{V}_T|+\log T)
\end{aligned}$$

Next, we bound the Term B. Similarly to the Term A, the Term B can be splitted as follows

$$\begin{aligned}
\sum_{t=2}^{T}Q_t &= \sum_{t=2}^{T}\big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2}-\alpha_t\widehat{V}_t^{-1/2}\big\|_2\\
&= \sum_{t=2}^{t_0}\big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2}-\alpha_t\widehat{V}_t^{-1/2}\big\|_2+\underbrace{\sum_{t=t_0+1}^{T}\big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2}-\alpha_t\widehat{V}_t^{-1/2}\big\|_2}_{T_2}
\end{aligned}$$

Also, in this case, the first term in RHS is independent of T, so we only bound T_2. The T_2 can be bound as

$$\begin{aligned}
\sum_{t=t_0+1}^{T} \big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2 &= \alpha \sum_{t=t_0+1}^{T} \Big\|\frac{1}{\sqrt{t-1}}\widehat{V}_{t-1}^{-1/2} - \frac{1}{\sqrt{t}}\widehat{V}_t^{-1/2}\Big\|_2 \\
&= \alpha \sum_{t=t_0+1}^{T} \big\|\widehat{G}_{t-1}^{-1/2} - \widehat{G}_t^{-1/2}\big\|_2 \\
&\leq \alpha \sum_{t=t_0+1}^{T} \mathrm{Tr}\big(\widehat{G}_{t-1}^{-1/2} - \widehat{G}_t^{-1/2}\big) \\
&= \alpha\big(\mathrm{Tr}(\widehat{G}_{t_0}^{-1/2}) - \mathrm{Tr}(\widehat{G}_T^{-1/2})\big) \\
&\leq \alpha \mathrm{Tr}(\widehat{G}_{t_0}^{-1/2}) = \mathcal{O}(1)
\end{aligned}$$

The remaining term is $\sum_{t=2}^{T-1} Q_t^2$ which can be derived from Term B with slight modifications.

$$\begin{aligned}
\sum_{t=2}^{T-1} Q_t^2 &= \sum_{t=2}^{T-1} \big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2^2 \\
&= \sum_{t=2}^{t_0} \big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2^2 + \underbrace{\sum_{t=t_0+1}^{T-1} \big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2^2}_{T_3}
\end{aligned}$$

Similarly to the Term B, we can bound T_3 with a little modification as

$$\begin{aligned}
\sum_{t=t_0+1}^{T-1} \big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2^2 &= \alpha^2 \sum_{t=t_0+1}^{T-1} \big\|\widehat{G}_{t-1}^{-1/2} - \widehat{G}_t^{-1/2}\big\|_2^2 \\
&\leq \alpha^2 \sum_{t=t_0+1}^{T-1} \mathrm{Tr}\Big(\big(\widehat{G}_{t-1}^{-1/2} - \widehat{G}_t^{-1/2}\big)^2\Big) \\
&\leq \alpha^2 \sum_{t=t_0+1}^{T-1} \mathrm{Tr}\Big(\big(\widehat{G}_{t-1}^{-1/2} - \widehat{G}_t^{-1/2}\big)\big(\widehat{G}_{t-1}^{-1/2} + \widehat{G}_t^{-1/2}\big)\Big) \\
&= \alpha^2 \sum_{t=t_0+1}^{T-1} \mathrm{Tr}\Big(\widehat{G}_{t-1}^{-1} - \widehat{G}_t^{-1}\Big) \\
&= \alpha^2 \mathrm{Tr}\big(\widehat{G}_{t_0}^{-1} - \widehat{G}_{T-1}^{-1}\big) \\
&\leq \alpha^2 \mathrm{Tr}\big(\widehat{G}_{t_0}^{-1}\big) = \mathcal{O}(1)
\end{aligned}$$

Therefore, we have $s_1(T) = \mathcal{O}\big(\log\big|\widehat{V}_T\big| + \log T + 1\big)$. Lastly, we should bound the LHS of (2).

$$\begin{aligned}
\mathbb{E}\Bigg[\sum_{t=1}^{T} \alpha_t \Big\langle \nabla f(x_t), \widehat{V}_t^{-1/2}\nabla f(x_t)\Big\rangle\Bigg] &\geq \mathbb{E}\Bigg[\sum_{t=1}^{T} \gamma_t \Big\|\nabla f(x_t)\Big\|^2\Bigg] \\
&\geq \mathbb{E}\Bigg[\min_{t\in[T]}\{\|\nabla f(x_t)\|^2\} \sum_{t=1}^{T} \gamma_t\Bigg]
\end{aligned}$$

Now, we bound the sum of γ_t.

$$\begin{aligned}\gamma_t &= \lambda_{\min}(\alpha_t \widehat{V}_t^{-1/2}) = \alpha\lambda_{\min}(G_t^{-1/2}) \\ &= \frac{\alpha}{\lambda_{\max}(G_t^{1/2})} \\ &= \frac{\alpha}{\lambda_{\max}(G_t)^{1/2}} \\ &\geq \frac{\alpha}{\Big(\sum\limits_{\tau=1}^{t} \|g_\tau\|^2\Big)^{1/2}} \geq \frac{\alpha}{\sqrt{T}G}\end{aligned}$$

From above observations, we have

$$\sum_{t=1}^{T} \gamma_t \geq \sum_{t=1}^{T} \frac{\alpha}{\sqrt{T}G} = \frac{\alpha\sqrt{T}}{G}$$

Therefore, we have $\Omega\big(s_2(T)\big) = \Omega(\sqrt{T})$. Finally, we conclude that

$$\min_{t\in[T]} \mathbb{E}\big[\|\nabla f(x_t)\|^2\big] = \mathcal{O}\left(\frac{\log\big|\widehat{V}_T\big| + \log T + 1}{\sqrt{T}}\right)$$

E.2 Proposition for RMSprop/Adam

Now, we can obtain *intuition* on the dynamics of Term A/Term B for EMA-based algorithms with the following proposition.

Proposition 1. *For block diagonal extensions of* RMSprop/Adam, *we set exponentially decaying stepsize* $\alpha_t = \alpha(\sqrt{\beta_2})^{t-1}$. *Then, the Term A* $\propto \log|\widehat{V}_T| - \log|\widehat{V}_{t_0}|$ *and the Term B is upper bounded with constant for any block sizes. Here,* t_0 *is the time when* $\widehat{V}_t$ *becomes full-rank. Hence, the term A/term B for EMA-based methods have the similar dynamics to those of* AdaGrad.

Proof. First, we will bound the Term A.

$$\sum_{t=1}^{T} \|\alpha_t \widehat{V}_t^{-1/2} g_t\|^2 = \sum_{t=1}^{t_0} \|\alpha_t \widehat{V}_t^{-1/2} g_t\|^2 + \underbrace{\sum_{t=t_0+1}^{T} \|\alpha_t \widehat{V}_t^{-1/2} g_t\|^2}_{T_1}$$

The first term in RHS is independent of T, so we only have to bound the term T_1. The Term T_1 can be bound as follows

$$\begin{aligned}T_1 &= \sum_{t=t_0+1}^{T} \|\alpha_t \widehat{V}_t^{-1/2} g_t\|^2 = \sum_{t=t_0+1}^{T} \|\alpha(\sqrt{\beta_2})^{t-1} \widehat{V}_t^{-1/2} g_t\|^2 \\ &\leq \sum_{t=t_0+1}^{T} \|\alpha \widehat{V}_t^{-1/2} g_t\|^2 = \alpha^2 \sum_{t=t_0+1}^{T} \|\widehat{V}_t^{-1/2} g_t\|^2 = \alpha^2 \sum_{t=t_0+1}^{T} \mathrm{Tr}(\widehat{V}_t^{-1} g_t g_t^T) \\ &\leq \frac{\alpha^2}{1-\beta_2} \sum_{t=t_0+1}^{T} \Big[\log|\widehat{V}_t| - \log|\beta_2 \widehat{V}_{t-1}|\Big] \\ &= \frac{\alpha^2}{1-\beta_2} \sum_{t=t_0+1}^{T} \Big[\log|\widehat{V}_t| - \log|\widehat{V}_{t-1}| + d\log\frac{1}{\beta_2}\Big] \\ &= \frac{\alpha^2}{1-\beta_2}\Big(\underbrace{\log|\widehat{V}_T| - \log|\widehat{V}_{t_0}|}_{\text{Dependent on } T} + d(T-t_0)\log\frac{1}{\beta_2}\Big)\end{aligned}$$

Summing up all the terms, we have

$$\sum_{t=1}^{T}\|\alpha_t\widehat{V}_t^{-1/2}g_t\|^2 \leq \sum_{t=1}^{t_0}\|\alpha_t\widehat{V}_t^{-1/2}g_t\|^2 + \frac{\alpha^2}{1-\beta_2}\Big(\underbrace{\log|\widehat{V}_T| - \log|\widehat{V}_{t_0}|}_{\text{Dependent on } T} + d(T-t_0)\log\frac{1}{\beta_2}\Big)$$

Now, we derive the bound for the Term B.

$$\begin{aligned}\sum_{t=2}^{T}Q_t &= \sum_{t=2}^{T}\big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2\\ &= \sum_{t=2}^{t_0}\big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2 + \sum_{t=t_0+1}^{T}\big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2\end{aligned}$$

As in the Term A, the first term in RHS is independent of T, and we can bound the second term as

$$\begin{aligned}\sum_{t=t_0+1}^{T}\big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2 &= \alpha\sum_{t=2}^{T}\big\|(\sqrt{\beta_2})^{t-2}\widehat{V}_{t-1}^{-1/2} - (\sqrt{\beta_2})^{t-1}\widehat{V}_t^{-1/2}\big\|_2\\ &= \alpha\sum_{t=t_0+1}^{T}(\sqrt{\beta_2})^{t-1}\Big\|\frac{1}{\sqrt{\beta_2}}\widehat{V}_{t-1}^{-1/2} - \widehat{V}_t^{-1/2}\Big\|_2\\ &= \alpha\sum_{t=t_0+1}^{T}(\sqrt{\beta_2})^{t-1}\big\|\big(\beta_2\widehat{V}_{t-1}\big)^{-1/2} - \widehat{V}_t^{-1/2}\big\|_2\\ &\leq \alpha\sum_{t=t_0+1}^{T}(\sqrt{\beta_2})^{t-1}\mathrm{Tr}\Big(\big(\beta_2\widehat{V}_{t-1}\big)^{-1/2} - \widehat{V}_t^{-1/2}\Big)\\ &= \alpha\sum_{t=t_0+1}^{T}\mathrm{Tr}\Big((\sqrt{\beta_2})^{t-2}\widehat{V}_{t-1}^{-1/2} - (\sqrt{\beta_2})^{t-1}\widehat{V}_t^{-1/2}\Big)\\ &= \alpha\Big(\mathrm{Tr}\big((\sqrt{\beta_2})^{t_0-1}\widehat{V}_{t_0}^{-1/2}\big) - \mathrm{Tr}\big((\sqrt{\beta_2})^{T-1}\widehat{V}_T^{-1/2}\big)\\ &\leq \alpha(\sqrt{\beta_2})^{t_0-1}\mathrm{Tr}\big(\widehat{V}_{t_0}^{-1/2}\big)\end{aligned}$$

Therefore, the bound for the Term B is independent of T.

$$\sum_{t=2}^{T}Q_t \leq \sum_{t=2}^{t_0}\big\|\alpha_{t-1}\widehat{V}_{t-1}^{-1/2} - \alpha_t\widehat{V}_t^{-1/2}\big\|_2 + \alpha(\sqrt{\beta_2})^{t_0-1}\mathrm{Tr}\big(\widehat{V}_{t_0}^{-1/2}\big) = \mathcal{O}(1)$$

As a result, we can expect that the Term A is related to the log-determinant of $\widehat{V}_T$ and the Term B is upper bounded as constant as in AdaGrad. □

F Proofs of Main Theorems

We study the following minimization problem,

$$\min f(x) := \mathbb{E}_\xi[f(x;\xi)]$$

under the assumption 1. The parameter x is an optimization variable, and ξ is a random variable representing randomly selected data sample from $\mathcal{D}$. We study the convergence analysis of the Algorithm 1. For analysis in stochastic convex optimization, one can refer to Duchi et al. (2011). For analysis in non-convex optimization with block diagonal (possibly full) matrix adaptations, we follow the arguments in the paper Chen et al. (2019). As we will show, the convergence of the adaptive block diagonal matrix adaptations depends on the changes of *effective spectrum* while the diagonal counterpart depends on the changes of *effective stepsize*. We assume that $\widehat{V}_t^{-1/2}$ means pseudo-inverse of $\widehat{V}_t^{1/2}$ if it is not full-rank. Note that, our proof can be applied to exact full matrix adaptations, Algorithm 1.

F.1 Technical Lemmas for Theorem 1

Lemma 3 (Generalized version of Lemma 6.1 in Chen et al. (2019))**.** *Consider the sequence*

$$z_t = x_t + \frac{\beta_{1,t}}{1-\beta_{1,t}}(x_t - x_{t-1})$$

Then, the following holds true

$$z_{t+1} - z_t = -\left(\frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} - \frac{\beta_{1,t}}{1-\beta_{1,t}}\right)\alpha_t \widehat{V}_t^{-1/2} m_t - \frac{\beta_{1,t}}{1-\beta_{1,t}}\left(\alpha_t \widehat{V}_t^{-1/2} - \alpha_{t-1}\widehat{V}_{t-1}^{-1/2}\right) m_{t-1} - \alpha_t \widehat{V}_t^{-1/2} g_t$$

Proof. By our update rule, we can derive

$$\begin{aligned}
x_{t+1} - x_t =& \; -\alpha_t \widehat{V}_t^{-1/2} m_t \\
\overset{(i)}{=}& \; -\alpha_t \widehat{V}_t^{-1/2}(\beta_{1,t} m_{t-1} + (1-\beta_{1,t}) g_t) \\
=& \; -\alpha_t \beta_{1,t} \widehat{V}_t^{-1/2} m_{t-1} - \alpha_t (1-\beta_{1,t}) \widehat{V}_t^{-1/2} g_t \\
\overset{(ii)}{=}& -\alpha_t \beta_{1,t} \widehat{V}_t^{-1/2}\Big(-\frac{1}{\alpha_{t-1}} \widehat{V}_{t-1}^{1/2}(x_t - x_{t-1})\Big) - \alpha_t(1-\beta_{1,t})\widehat{V}_t^{-1/2} g_t \\
=& \; \frac{\alpha_t}{\alpha_{t-1}} \beta_{1,t} (\widehat{V}_t^{-1}\widehat{V}_{t-1})^{1/2}(x_t - x_{t-1}) - \alpha_t(1-\beta_{1,t})\widehat{V}_t^{-1/2} g_t \\
=& \; \beta_{1,t}(x_t - x_{t-1}) + \beta_{1,t}\left(\frac{\alpha_t}{\alpha_{t-1}}(\widehat{V}_t^{-1}\widehat{V}_{t-1})^{1/2} - I_d\right)(x_t - x_{t-1}) - \alpha_t(1-\beta_{1,t})\widehat{V}_t^{-1/2} g_t \\
\overset{(iii)}{=}& \; \beta_{1,t}(x_t - x_{t-1}) - \beta_{1,t}\left(\alpha_t \widehat{V}_t^{-1/2} - \alpha_{t-1}\widehat{V}_{t-1}^{-1/2}\right) m_{t-1} - \alpha_t(1-\beta_{1,t})\widehat{V}_t^{-1/2} g_t
\end{aligned}$$

The reasoning follows

(i) By definition of m_t.

(ii) Since $x_t = x_{t-1} - \alpha_{t-1}\widehat{V}_{t-1}^{-1/2} m_{t-1}$, we can solve as $m_{t-1} = -\frac{1}{\alpha_{t-1}} \widehat{V}_{t-1}^{1/2}(x_t - x_{t-1})$.

(iii) Similarly to (ii), we can have $\widehat{V}_{t-1}^{1/2}(x_t - x_{t-1})/\alpha_{t-1} = -m_{t-1}$.

Since $x_{t+1} - x_t = (1-\beta_{1,t})x_{t+1} + \beta_{1,t}(x_{t+1} - x_t) - (1-\beta_{1,t})x_t$, we can further derive by combining the above,

$$\begin{aligned}
& (1-\beta_{1,t})x_{t+1} + \beta_{1,t}(x_{t+1} - x_t) \\
=& \; (1-\beta_{1,t})x_t + \beta_{1,t}(x_t - x_{t-1}) - \beta_{1,t}\left(\alpha_t \widehat{V}_t^{-1/2} - \alpha_{t-1}\widehat{V}_{t-1}^{-1/2}\right) m_{t-1} - \alpha_t(1-\beta_{1,t})\widehat{V}_t^{-1/2} g_t
\end{aligned}$$

By dividing both sides by $1-\beta_{1,t}$,

$$\begin{aligned}
& x_{t+1} + \frac{\beta_{1,t}}{1-\beta_{1,t}}(x_{t+1} - x_t) \\
=& \; x_t + \frac{\beta_{1,t}}{1-\beta_{1,t}}(x_t - x_{t-1}) - \frac{\beta_{1,t}}{1-\beta_{1,t}}\left(\alpha_t \widehat{V}_t^{-1/2} - \alpha_{t-1}\widehat{V}_{t-1}^{-1/2}\right) m_{t-1} - \alpha_t \widehat{V}_t^{-1/2} g_t
\end{aligned}$$

Define the sequence

$$z_t = x_t + \frac{\beta_{1,t}}{1-\beta_{1,t}}(x_t - x_{t-1})$$

Then, we obtain

$$
\begin{aligned}
z_{t+1} &= z_t + \left(\frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} - \frac{\beta_{1,t}}{1-\beta_{1,t}}\right)(x_{t+1} - x_t) \\
&\quad - \frac{\beta_{1,t}}{1-\beta_{1,t}}\left(\alpha_t \widehat{V}_t^{-1/2} - \alpha_{t-1}\widehat{V}_{t-1}^{-1/2}\right)m_{t-1} - \alpha_t \widehat{V}_t^{-1/2} g_t \\
&= z_t - \left(\frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} - \frac{\beta_{1,t}}{1-\beta_{1,t}}\right)\alpha_t \widehat{V}_t^{-1/2} m_t \\
&\quad - \frac{\beta_{1,t}}{1-\beta_{1,t}}\left(\alpha_t \widehat{V}_t^{-1/2} - \alpha_{t-1}\widehat{V}_{t-1}^{-1/2}\right)m_{t-1} - \alpha_t \widehat{V}_t^{-1/2} g_t
\end{aligned}
$$

By putting z_t to the left hand side, we can get desired relations. □

Lemma 4 (Generalized version of Lemma 6.2 in Chen et al. (2019))**.** *Suppose that the assumptions in Theorem 1 hold, then*

$$
\mathbb{E}[f(z_{t+1}) - f(z_1)] \leq \sum_{i=1}^{6} T_i
$$

where

$$
\begin{aligned}
T_1 &= -\mathbb{E}\left[\sum_{i=1}^{t}\left\langle \nabla f(z_i), \frac{\beta_{1,i}}{1-\beta_{1,i}}\left(\alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1}\widehat{V}_{i-1}^{-1/2}\right)m_{i-1}\right\rangle\right] \\
T_2 &= -\mathbb{E}\left[\sum_{i=1}^{t}\alpha_i\left\langle \nabla f(z_i), \widehat{V}_i^{-1/2} g_i\right\rangle\right] \\
T_3 &= -\mathbb{E}\left[\sum_{i=1}^{t}\left\langle \nabla f(z_i), \left(\frac{\beta_{1,i+1}}{1-\beta_{1,i+1}} - \frac{\beta_{1,i}}{1-\beta_{1,i}}\right)\alpha_i \widehat{V}_i^{-1/2} m_i\right\rangle\right] \\
T_4 &= \mathbb{E}\left[\sum_{i=1}^{t}\frac{3}{2}L\left\|\left(\frac{\beta_{1,i+1}}{1-\beta_{1,i+1}} - \frac{\beta_{1,i}}{1-\beta_{1,i}}\right)\alpha_i \widehat{V}_i^{-1/2} m_i\right\|^2\right] \\
T_5 &= \mathbb{E}\left[\sum_{i=1}^{t}\frac{3}{2}L\left\|\frac{\beta_{1,i}}{1-\beta_{1,i}}\left(\alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1}\widehat{V}_{i-1}^{-1/2}\right)m_{i-1}\right\|^2\right] \\
T_6 &= \mathbb{E}\left[\sum_{i=1}^{t}\frac{3}{2}L\left\|\alpha_i \widehat{V}_i^{-1/2} g_i\right\|^2\right]
\end{aligned}
$$

Proof. By L-Lipschitz continuous gradients, we get the following quadratic upper bound,

$$
f(z_{t+1}) \leq f(z_t) + \langle \nabla f(z_t), z_{t+1} - z_t\rangle + \frac{L}{2}\|z_{t+1} - z_t\|^2
$$

Let $d_t = z_{t+1} - z_t$. The lemma 2 yields

$$
d_t = -\left(\frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} - \frac{\beta_{1,t}}{1-\beta_{1,t}}\right)\alpha_t V_t^{-1/2} m_t - \frac{\beta_{1,t}}{1-\beta_{1,t}}\left(\alpha_t V_t^{-1/2} - \alpha_{t-1}V_{t-1}^{-1/2}\right)m_{t-1} - \alpha_t V_t^{-1/2} g_t
$$

Combining with Lipschitz continuous gradients, we have

$$\begin{aligned}\mathbb{E}[f(z_{t+1})-f(z_1)] &= \mathbb{E}\left[\sum_{i=1}^{t} f(z_{i+1})-f(z_i)\right]\\ &\leq \mathbb{E}\left[\sum_{i=1}^{t}\langle\nabla f(z_i), d_i\rangle + \frac{L}{2}\|d_i\|^2\right]\\ &= -\mathbb{E}\left[\sum_{i=1}^{t}\left\langle \nabla f(z_i), \frac{\beta_{1,i}}{1-\beta_{1,i}}\left(\alpha_i V_i^{-1/2}-\alpha_{i-1}V_{i-1}^{-1/2}\right)m_{i-1}\right\rangle\right]\\ &\quad -\mathbb{E}\left[\sum_{i=1}^{t}\alpha_i\left\langle\nabla f(z_i), V_i^{-1/2}g_i\right\rangle\right]\\ &\quad -\mathbb{E}\left[\sum_{i=1}^{t}\left\langle\nabla f(z_i), \left(\frac{\beta_{1,i+1}}{1-\beta_{1,i+1}}-\frac{\beta_{1,i}}{1-\beta_{1,i}}\right)\alpha_i V_i^{-1/2}m_i\right\rangle\right]\\ &\quad +\mathbb{E}\left[\sum_{i=1}^{t}\frac{L}{2}\|d_i\|^2\right] = T_1+T_2+T_3+\mathbb{E}\left[\sum_{i=1}^{t}\frac{L}{2}\|d_i\|^2\right]\end{aligned}$$

With $\|a+b+c\|^2 \leq 3(\|a\|^2+\|b\|^2+\|c\|^2)$, we can finally bound by

$$\mathbb{E}[f(z_{t+1})-f(z_1)] \leq \sum_{i=1}^{6} T_i$$

□

Lemma 5 (Generalized version of Lemma 6.3 in Chen et al. (2019)). *Suppose that the assumptions in Theorem 1 hold, T_1 can be bound as*

$$T_1 \leq G^2\frac{\beta_1}{1-\beta_1}\mathbb{E}\left[\sum_{i=1}^{t}\left\|\alpha_i V_i^{-1/2}-\alpha_{i-1}V_{i-1}^{-1/2}\right\|_2\right]$$

Proof. From the definition of quantity T_1,

$$\begin{aligned}T_1 = &\ -\mathbb{E}\left[\sum_{i=1}^{t}\left\langle\nabla f(z_i), \frac{\beta_{1,i}}{1-\beta_{1,i}}\left(\alpha_i V_i^{-1/2}-\alpha_{i-1}V_{i-1}^{-1/2}\right)m_{i-1}\right\rangle\right]\\ \overset{(i)}{\leq} &\ \mathbb{E}\left[\sum_{i=1}^{t}\|\nabla f(z_i)\|_2\left\|\frac{\beta_{1,i}}{1-\beta_{1,i}}\left(\alpha_i V_i^{-1/2}-\alpha_{i-1}V_{i-1}^{-1/2}\right)m_{i-1}\right\|_2\right]\\ \overset{(ii)}{\leq} &\ \frac{\beta_1}{1-\beta_1}\mathbb{E}\left[\sum_{i=1}^{t}\|\nabla f(z_i)\|_2\left\|\alpha_i V_i^{-1/2}-\alpha_{i-1}V_{i-1}^{-1/2}\right\|_2\|m_{t-1}\|_2\right]\\ \overset{(iii)}{\leq} &\ G^2\frac{\beta_1}{1-\beta_1}\mathbb{E}\left[\sum_{i=1}^{t}\left\|\alpha_i V_i^{-1/2}-\alpha_{i-1}V_{i-1}^{-1/2}\right\|_2\right]\end{aligned}$$

The reasoning follows

(i) By Cauchy-Schwarz inequality.

(ii) For a matrix norm, we have $\|Ax\|_2 \leq \|A\|_2\|x\|_2$. Also, $\frac{\beta_{1,i}}{1-\beta_{1,i}} = \frac{1}{1-\beta_{1,i}} - 1 \leq \frac{1}{1-\beta_1} - 1 = \frac{\beta_1}{1-\beta_1}$.

(iii) By definition of m_t, we have $m_t = \beta_{1,t}m_{t-1} + (1-\beta_{1,t})g_t$. Therefore, we use a triangle inequality by $\|m_t\|_2 \leq \beta_{1,t}\|m_{t-1}\|_2 + (1-\beta_1)\|g_t\|_2 \leq (\beta_{1,t}+1-\beta_{1,t})\max\{\|m_{t-1}\|_2, \|g_t\|_2\}$. Since we have $m_0 = 0$ and $\|g_t\| \leq G$, we also have $\|m_t\| \leq G$ by the mathematical induction.

□

Lemma 6 (Generalized version of Lemma 6.4 in Chen et al. (2019))**.** *Suppose that the assumptions in Theorem 1 hold, then T_3 can be bound as*

$$T_3 \le \left(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}}\right)(G^2 + D^2)$$

Proof. By the definition of T_3,

$$\begin{aligned} T_3 = &\; -\mathbb{E}\left[\sum_{i=1}^{t}\left\langle \nabla f(z_i), \left(\frac{\beta_{1,i+1}}{1-\beta_{1,i+1}} - \frac{\beta_{1,i}}{1-\beta_{1,i}}\right)\alpha_i V_i^{-1/2} m_i\right\rangle\right] \\ \overset{(i)}{\le} &\; \mathbb{E}\left[\sum_{i=1}^{t}\left|\frac{\beta_{1,i+1}}{1-\beta_{1,i+1}} - \frac{\beta_{1,i}}{1-\beta_{1,i}}\right|\frac{1}{2}\Big(\|\nabla f(z_i)\|^2 + \|\alpha_i V_i^{-1/2} m_i\|^2\Big)\right] \\ \overset{(ii)}{\le} &\; \mathbb{E}\left[\sum_{i=1}^{t}\left|\frac{\beta_{1,i+1}}{1-\beta_{1,i+1}} - \frac{\beta_{1,i}}{1-\beta_{1,i}}\right|\frac{1}{2}\Big(G^2 + D^2\Big)\right] \\ = &\; \sum_{i=1}^{t}\left(\frac{\beta_{1,i}}{1-\beta_{1,i}} - \frac{\beta_{1,i+1}}{1-\beta_{1,i+1}}\right)\frac{1}{2}\Big(G^2 + D^2\Big) \\ \overset{(iii)}{\le} &\; \left(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}}\right)(G^2 + D^2) \end{aligned}$$

The reasoning follows

(i) Use Cauchy-Schwarz inequality and $ab \le \frac{1}{2}(a^2 + b^2)$ for $a, b \ge 0$.

(ii) By our assumptions on bounded gradients and bounded final step vectors.

(iii) The sum over $i = 1$ to T can be done by telescoping.

□

Lemma 7 (Generalized version of Lemma 6.5 in Chen et al. (2019))**.** *Suppose that the assumptions in Theorem 1 hold, T_4 can be bound as*

$$T_4 \le \Big(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}}\Big)^2 D^2$$

Proof. By the definition of T_4,

$$\begin{aligned} \frac{2}{3L}T_4 = &\; \mathbb{E}\left[\sum_{i=1}^{t}\left\|\left(\frac{\beta_{1,i+1}}{1-\beta_{1,i+1}} - \frac{\beta_{1,i}}{1-\beta_{1,i}}\right)\alpha_i V_i^{-1/2} m_i\right\|^2\right] \\ \overset{(i)}{\le} &\; \mathbb{E}\left[\sum_{i=1}^{t}\Big(\frac{\beta_{1,i+1}}{1-\beta_{1,i+1}} - \frac{\beta_{1,i}}{1-\beta_{1,i}}\Big)^2 D^2\right] \\ \overset{(ii)}{\le} &\; \Big(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}}\Big)\sum_{i=1}^{t}\Big(\frac{\beta_{1,i+1}}{1-\beta_{1,i+1}} - \frac{\beta_{1,i}}{1-\beta_{1,i}}\Big)D^2 \\ \overset{(iii)}{\le} &\; \Big(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}}\Big)^2 D^2 \end{aligned}$$

The reasoning follows

(i) From our assumptions on final step vector $\|\alpha_i \widehat{V}_i^{-1/2} m_i\|^2 \le D$.

(ii) We use the relation $\beta_1 \geq \beta_{1,t} \leq \beta_{1,t+1}$.

(iii) By telescoping sum, we can get the final result.

□

Lemma 8 (Generalized version of Lemma 6.6 in Chen et al. (2019)). *Suppose that the assumptions in Theorem 1 hold, T_5 can be bound as*

$$\frac{2}{3L}T_5 \leq \Big(\frac{\beta_1}{1-\beta_1}\Big)^2 G^2 \mathbb{E}\left[\sum_{i=2}^{t} \left|\!\left|\!\left| \alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1}\widehat{V}_{i-1}^{-1/2} \right|\!\right|\!\right|_2\right]$$

Proof. By the definition of T_5,

$$\begin{aligned}
\frac{2}{3L}T_5 =\ & \mathbb{E}\left[\sum_{i=2}^{t} \left\| \frac{\beta_{1,i}}{1-\beta_{1,i}} \left(\alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2}\right) m_{i-1} \right\|^2\right] \\
\overset{(i)}{\leq}\ & \mathbb{E}\left[\sum_{i=2}^{t} \frac{\beta_{1,i}}{1-\beta_{1,i}} \left|\!\left|\!\left| \alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1}\widehat{V}_{i-1}^{-1/2} \right|\!\right|\!\right|_2^2 \|m_{i-1}\|_2^2\right] \\
\overset{(ii)}{\leq}\ & \Big(\frac{\beta_1}{1-\beta_1}\Big)^2 G^2 \mathbb{E}\left[\sum_{i=2}^{t} \left|\!\left|\!\left| \alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1}\widehat{V}_{i-1}^{-1/2} \right|\!\right|\!\right|_2^2\right]
\end{aligned}$$

The reasoning follows

(i) By the matrix norm inequality, we use $\|Ax\|_2 \leq |\!|\!|A|\!|\!|_2 \|x\|_2$.

(ii) We can obtain the result using $\beta_1 \geq \beta_{1,t} \geq \beta_{1,t+1}$.

□

Lemma 9 (Generalized version of Lemma 6.7 in Chen et al. (2019)). *Suppose that the assumptions in Theorem 1 hold, The quantity T_2 can be bound as*

$$\begin{aligned}
T_2 \leq\ & L^2\Big(\frac{\beta_1}{1-\beta_1}\Big)^2 T_8 + L^2\Big(\frac{\beta_1}{1-\beta_1}\Big)^2 T_9 + \frac{1}{2}\mathbb{E}\left[\sum_{i=1}^{t} \|\alpha_i \widehat{V}_i^{-1/2} g_i\|^2\right] \\
& + 2G^2\mathbb{E}\left[\sum_{i=2}^{t} \left|\!\left|\!\left| \alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2} \right|\!\right|\!\right|_2\right] + 2G^2\mathbb{E}\left[\left|\!\left|\!\left| \alpha_1 V_1^{-1/2} \right|\!\right|\!\right|_2\right] \\
& - \mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2}\nabla f(x_i) \right\rangle\right]
\end{aligned}$$

Proof. First, note that,

$$z_i - x_i = \frac{\beta_{1,i}}{1-\beta_{1,i}}(x_i - x_{i-1}) = -\frac{\beta_{1,i}}{1-\beta_{1,i}}\alpha_{i-1}\widehat{V}_{i-1}^{-1/2} m_{i-1}$$

By the definition of T_2 and $z_1 = x_1$, we have

$$\begin{aligned}
T_2 &= -\mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(z_i), \widehat{V}_i^{-1/2} g_i \right\rangle\right] \\
&= -\mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), \widehat{V}_i^{-1/2} g_i \right\rangle\right] - \mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(z_i) - \nabla f(x_i), \widehat{V}_i^{-1/2} g_i \right\rangle\right]
\end{aligned}$$

The second term can be bounded as

$$
\begin{aligned}
&-\mathbb{E}\Bigg[\sum_{i=1}^{t}\alpha_i\bigg\langle \nabla f(z_i)-\nabla f(x_i), \widehat{V}_i^{-1/2}g_i\bigg\rangle\Bigg] \\
&\overset{(i)}{\le} \mathbb{E}\Bigg[\sum_{i=1}^{t}\frac{1}{2}\|\nabla f(z_i)-\nabla f(x_i)\|^2+\frac{1}{2}\|\alpha_i\widehat{V}_i^{-1/2}g_i\|^2\Bigg] \\
&\overset{(ii)}{\le} \frac{L^2}{2}T_7+\frac{1}{2}\mathbb{E}\Bigg[\sum_{i=1}^{t}\|\alpha_i\widehat{V}_i^{-1/2}g_i\|^2\Bigg]
\end{aligned}
$$

(i) is due to Cauchy-Schwarz inequality and $ab \le \frac{1}{2}a^2+\frac{1}{2}b^2$ for $a, b \ge 0$. (ii) is as follows: By L-Lipschitz continuous gradients, we have

$$
\|\nabla f(z_i)-\nabla f(x_i)\| \le L\|z_i-x_i\| = L\bigg\|\frac{\beta_{1,t}}{1-\beta_{1,t}}\alpha_{i-1}V_{i-1}^{-1/2}m_{i-1}\bigg\|
$$

Let T_7 be

$$
T_7=\mathbb{E}\Bigg[\sum_{i=1}^{t}\bigg\|\frac{\beta_{1,i}}{1-\beta_{1,i}}\alpha_{i-1}V_{i-1}^{-1/2}m_{i-1}\bigg\|^2\Bigg]
$$

We should bound the quantity T_7, by the definition of m_t, we have

$$
m_i=\sum_{k=1}^{i}\big[\big(\prod_{l=k+1}^{i}\beta_{1,l}\big)(1-\beta_{1,k})g_k\big]
$$

Plugging m_{i-1} into T_7 yields

$$
\begin{aligned}
T_7 =\ & \mathbb{E}\Bigg[\sum_{i=1}^{t}\bigg\|\frac{\beta_{1,i}}{1-\beta_{1,i}}\alpha_{i-1}V_{i-1}^{-1/2}m_{i-1}\bigg\|^2\Bigg] \\
\overset{(i)}{\le}\ & \Big(\frac{\beta_1}{1-\beta_1}\Big)^2\mathbb{E}\Bigg[\sum_{i=2}^{t}\bigg\|\alpha_{i-1}V_{i-1}^{-1/2}\sum_{k=1}^{i-1}\Big[\Big(\prod_{l=k+1}^{i-1}\beta_{1,l}\Big)(1-\beta_{1,k})g_k\Big]\bigg\|^2\Bigg] \\
=\ & \Big(\frac{\beta_1}{1-\beta_1}\Big)^2\mathbb{E}\Bigg[\sum_{i=2}^{t}\bigg\|\sum_{k=1}^{i-1}\alpha_{i-1}V_{i-1}^{-1/2}\Big[\Big(\prod_{l=k+1}^{i-1}\beta_{1,l}\Big)(1-\beta_{1,k})g_k\Big]\bigg\|^2\Bigg] \\
\overset{(ii)}{\le}\ & 2\Big(\frac{\beta_1}{1-\beta_1}\Big)^2\underbrace{\mathbb{E}\Bigg[\sum_{i=2}^{t}\bigg\|\sum_{k=1}^{i-1}\alpha_kV_k^{-1/2}\Big[\Big(\prod_{l=k+1}^{i-1}\beta_{1,l}\Big)(1-\beta_{1,k})g_k\Big]\bigg\|^2\Bigg]}_{T_8} \\
& +2\Big(\frac{\beta_1}{1-\beta_1}\Big)^2\underbrace{\mathbb{E}\Bigg[\sum_{i=2}^{t}\bigg\|\sum_{k=1}^{i-1}\Big(\alpha_iV_i^{-1/2}-\alpha_kV_k^{-1/2}\Big)\Big[\Big(\prod_{l=k+1}^{i-1}\beta_{1,l}\Big)(1-\beta_{1,k})g_k\Big]\bigg\|^2\Bigg]}_{T_9}
\end{aligned}
$$

(i) is by $\beta_1 \geq \beta_{1,t}$ and (ii) is by We use the fact $(a+b) \leq 2(\|a\|^2 + \|b\|^2)$ in (i). We first bound T_8 as below

$$\begin{aligned}
T_8 &= \mathbb{E}\Bigg[\sum_{i=2}^{t}\bigg\|\sum_{k=1}^{i-1}\alpha_k V_k^{-1/2}\Big[\Big(\prod_{l=k+1}^{i-1}\beta_{1,l}\Big)(1-\beta_{1,k})g_k\Big]\bigg\|^2\Bigg] \\
&= \mathbb{E}\Bigg[\sum_{i=2}^{t}\sum_{j=1}^{d}\Bigg(\sum_{k=1}^{i-1}\alpha_k V_k^{-1/2}\Big[\Big(\prod_{l=k+1}^{i-1}\beta_{1,l}\Big)(1-\beta_{1,k})g_k\Big]\Bigg)_j^2\Bigg] \\
&= \mathbb{E}\Bigg[\sum_{i=2}^{t}\sum_{j=1}^{d}\Bigg(\sum_{k=1}^{i-1}\sum_{p=1}^{i-1}\Big(\alpha_k V_k^{-1/2}g_k\Big)_j\Big(\prod_{l=k+1}^{i-1}\beta_{1,l}\Big)(1-\beta_{1,k})\Big(\alpha_p V_p^{-1/2}g_p\Big)_j\Big(\prod_{q=p+1}^{i-1}\beta_{1,q}\Big)(1-\beta_{1,p})\Bigg)\Bigg] \\
&\leq \mathbb{E}\Bigg[\sum_{i=2}^{t}\sum_{j=1}^{d}\Bigg(\sum_{k=1}^{i-1}\sum_{p=1}^{i-1}(\beta_1^{i-1-k})(\beta_1^{i-1-p})\frac{1}{2}\bigg\{\Big(\alpha_k V_k^{-1/2}g_k\Big)_j^2 + \Big(\alpha_p V_p^{-1/2}g_p\Big)_j^2\bigg\}\Bigg)\Bigg] \\
&= \mathbb{E}\Bigg[\sum_{i=2}^{t}\sum_{j=1}^{d}\Bigg(\sum_{k=1}^{i-1}(\beta_1^{i-1-k})\Big(\alpha_k V_k^{-1/2}g_k\Big)_j^2\sum_{p=1}^{i-1}(\beta_1^{i-1-p})\Bigg)\Bigg] \\
&\leq \frac{1}{1-\beta_1}\mathbb{E}\Bigg[\sum_{i=2}^{t}\sum_{j=1}^{d}\sum_{k=1}^{i-1}(\beta_1^{i-1-k})\Big(\alpha_k V_k^{-1/2}g_k\Big)_j^2\Bigg] \\
&= \frac{1}{1-\beta_1}\mathbb{E}\Bigg[\sum_{k=1}^{t-1}\sum_{j=1}^{d}\sum_{i=k+1}^{t}(\beta_1^{i-1-k})\Big(\alpha_k V_k^{-1/2}g_k\Big)_j^2\Bigg] \\
&= \Big(\frac{1}{1-\beta_1}\Big)^2\mathbb{E}\Bigg[\sum_{k=1}^{t-1}\sum_{j=1}^{d}\Big(\alpha_k V_k^{-1/2}g_k\Big)_j^2\Bigg] = \Big(\frac{1}{1-\beta_1}\Big)^2\mathbb{E}\Bigg[\sum_{i=1}^{t}\Big\|\alpha_i V_i^{-1/2}g_i\Big\|^2\Bigg]
\end{aligned}$$

For the T_9 bound, we have

$$\begin{aligned}
T_9 &= \mathbb{E}\Bigg[\sum_{i=2}^{t}\bigg\|\sum_{k=1}^{i-1}\Big[\Big(\prod_{l=k+1}^{i-1}\beta_{1,l}\Big)(1-\beta_{1,k})\Big]\Big(\alpha_i V_i^{-1/2} - \alpha_k V_k^{-1/2}\Big)g_k\bigg\|^2\Bigg] \\
&\leq \mathbb{E}\Bigg[\sum_{i=2}^{t}\Bigg(\sum_{k=1}^{i-1}\Big[\Big(\prod_{l=k+1}^{i-1}\beta_{1,l}\Big)(1-\beta_{1,k})\Big]\bigg\|\alpha_i V_i^{-1/2} - \alpha_k V_k^{-1/2}\bigg\|_2\|g_k\|_2\Bigg)^2\Bigg] \\
&\leq \mathbb{E}\Bigg[\sum_{i=1}^{t-1}\Bigg(\sum_{k=1}^{i}\Big[\Big(\prod_{l=k+1}^{i}\beta_{1,l}\Big)\Big]\bigg\|\alpha_i V_i^{-1/2} - \alpha_k V_k^{-1/2}\bigg\|_2\|g_k\|_2\Bigg)^2\Bigg] \\
&\leq G^2\mathbb{E}\Bigg[\sum_{i=1}^{t-1}\Bigg(\sum_{k=1}^{i}\beta_1^{i-k}\bigg\|\alpha_i V_i^{-1/2} - \alpha_k V_k^{-1/2}\bigg\|_2\Bigg)^2\Bigg] \\
&\leq G^2\mathbb{E}\Bigg[\sum_{i=1}^{t-1}\Bigg(\sum_{k=1}^{i}\beta_1^{i-k}\sum_{l=k+1}^{i}\bigg\|\alpha_l V_l^{-1/2} - \alpha_{l-1} V_{l-1}^{-1/2}\bigg\|_2\Bigg)^2\Bigg] \\
&\leq G^2\Big(\frac{1}{1-\beta_1}\Big)^2\Big(\frac{\beta_1}{1-\beta_1}\Big)^2\mathbb{E}\Bigg[\sum_{i=2}^{t-1}\bigg\|\alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2}\bigg\|_2^2\Bigg]
\end{aligned}$$

Then, the remaining term is

$$\mathbb{E}\Bigg[\sum_{i=1}^{t}\alpha_i\Big\langle\nabla f(x_i), V_i^{-1/2}g_i\Big\rangle\Bigg]$$

To find the upper bound for this term, we reparameterize $g_t = \nabla f(x_t) + \delta_t$ with $\mathbb{E}[\delta_t] = 0$, and we have

$$\begin{aligned}
&\mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} g_i \right\rangle\right] \\
=\ &\mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} (\nabla f(x_i) + \delta_i) \right\rangle\right] \\
=\ &\mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} \nabla f(x_i) \right\rangle\right] + \left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} \delta_i \right\rangle\right]
\end{aligned}$$

For the second term of last equation,

$$\begin{aligned}
&\mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} \delta_i \right\rangle\right] \\
=\ &\mathbb{E}\left[\sum_{i=2}^{t} \left\langle \nabla f(x_i), \left(\alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2}\right) \delta_i \right\rangle\right] + \mathbb{E}\left[\sum_{i=2}^{t} \alpha_{i-1} \left\langle \nabla f(x_i), V_{i-1}^{-1/2} \delta_i \right\rangle\right] + \mathbb{E}\left[\alpha_1 \left\langle \nabla f(x_1), V_1^{-1/2} \delta_1 \right\rangle\right] \\
=\ &\mathbb{E}\left[\sum_{i=2}^{t} \left\langle \nabla f(x_i), \left(\alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2}\right) \delta_i \right\rangle\right] + \mathbb{E}\left[\alpha_1 \nabla f(x_1)^T V_1^{-1/2} \delta_1\right] \\
\overset{(i)}{\geq}\ &\mathbb{E}\left[\sum_{i=2}^{t} \left\langle \nabla f(x_i), \left(\alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2}\right) \delta_i \right\rangle\right] - 2G^2 \mathbb{E}\left[\left\vert\left\vert\left\vert \alpha_1 V_1^{-1/2} \right\vert\right\vert\right\vert_2\right]
\end{aligned}$$

The reasoning is as follows:

(i) The conditional expectation $\mathbb{E}\left[V_{i-1}^{-1/2} \delta_i \middle| x_i, \widehat{V}_{i-1}\right] = 0$ since the $\widehat{V}_{i-1}$ only depends on the noise variables $\xi_1, \cdots, \xi_{i-1}$ and δ_i depends on ξ_i with $\mathbb{E}[\xi_k] = 0$ for all $k \in \{1, 2, ..., i\}$. Therefore, they are independent.

Further, we have

$$\begin{aligned}
\mathbb{E}\left[\sum_{i=2}^{t} \left\langle \nabla f(x_i), \left(\alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2}\right) \delta_i \right\rangle\right] &\geq -\mathbb{E}\left[\sum_{i=2}^{t} \left| \left\langle \nabla f(x_i), \left(\alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2}\right) \delta_i \right\rangle \right|\right] \\
&\overset{(ii)}{\geq} -\mathbb{E}\left[\sum_{i=2}^{t} \left\|\nabla f(x_i)\right\|_2 \left\|\left(\alpha_i V_i^{-1/2} - \alpha_{i-1}^{-1/2}\right) \delta_i\right\|_2\right] \\
&\overset{(iii)}{\geq} -2G^2 \mathbb{E}\left[\sum_{i=2}^{t} \left\vert\left\vert\left\vert \alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2} \right\vert\right\vert\right\vert_2\right]
\end{aligned}$$

Therefore, we can bound the first term

$$\begin{aligned}
&-\mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} g_i \right\rangle\right] \\
\leq\ &2G^2 \mathbb{E}\left[\sum_{i=2}^{t} \left\vert\left\vert\left\vert \alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2} \right\vert\right\vert\right\vert_2\right] + 2G^2 \mathbb{E}\left[\left\vert\left\vert\left\vert \alpha_1 V_1^{-1/2} \right\vert\right\vert\right\vert_2\right] - \mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} \nabla f(x_i) \right\rangle\right]
\end{aligned}$$

□

Lemma 10 (Lemma 6.8 in Chen et al. (2019)). *For $a_i \leq 0$, $\beta \in [0, 1)$, and $b_i = \sum_{k=1}^{i} \beta^{i-k} \sum_{l=k+1}^{i} a_l$, we have*

$$\sum_{i=1}^{t} b_i^2 \leq \left(\frac{1}{1-\beta}\right)^2 \left(\frac{\beta}{1-\beta}\right)^2 \sum_{i=2}^{t} a_i^2$$

F.2 Proof of Theorem 1

Proof. We combine the above lemmas to bound

$$
\begin{aligned}
\mathbb{E}[f(z_{t+1}) - f(z_1)] &\leq \sum_{i=1}^{6} T_i \\
&\leq \underbrace{G^2 \frac{\beta_1}{1-\beta_1} \mathbb{E}\left[\sum_{i=2}^{t} \left|\!\left|\!\left| \alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2} \right|\!\right|\!\right|_2 \right]}_{T_1} \\
&\quad \underbrace{+ \left(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} \right) (G^2 + D^2)}_{T_3} \\
&\quad \underbrace{+ \Big(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} \Big)^2 D^2}_{T_4} \\
&\quad \underbrace{+ \Big(\frac{\beta_1}{1-\beta_1} \Big)^2 G^2 \mathbb{E}\left[\sum_{i=2}^{t} \left|\!\left|\!\left| \alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2} \right|\!\right|\!\right|_2^2 \right]}_{T_5} \\
&\quad \underbrace{+ \mathbb{E}\left[\sum_{i=1}^{t} \frac{3}{2} L \left\| \alpha_i V_i^{-1/2} g_i \right\|^2 \right]}_{T_6} \\
&\quad \underbrace{+ 2G^2 \mathbb{E}\left[\sum_{i=2}^{t} \left|\!\left|\!\left| \alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2} \right|\!\right|\!\right|_2 \right] + 2G^2 \mathbb{E}\left[\left|\!\left|\!\left| \alpha_1 V_1^{-1/2} \right|\!\right|\!\right|_2 \right]}_{T_2} \\
&\quad \underbrace{- \mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} \nabla f(x_i) \right\rangle \right]}_{T_2} \\
&\quad \underbrace{+ L^2 \Big(\frac{\beta_1}{1-\beta_1} \Big)^2 \left(\Big(\frac{1}{1-\beta_1} \Big)^2 \mathbb{E}\left[\sum_{i=1}^{t-1} \left\| \alpha_i V_i^{-1/2} g_i \right\|^2 \right] \right.}_{T_2} \\
&\quad \underbrace{\left. + G^2 \Big(\frac{1}{1-\beta_1} \Big)^2 \Big(\frac{\beta_1}{1-\beta_1} \Big)^2 \mathbb{E}\left[\sum_{i=2}^{t-1} \left|\!\left|\!\left| \alpha_i V_i^{-1/2} - \alpha_{i-1} V_{i-1}^{-1/2} \right|\!\right|\!\right|_2^2 \right] \right)}_{T_2} \\
&\quad \underbrace{+ \mathbb{E}\left[\frac{1}{2} \sum_{i=1}^{t} \| \alpha_i V_i^{-1/2} g_i \|^2 \right]}_{T_2}
\end{aligned}
$$

By merging similar terms, we can have

$$\begin{aligned}
\mathbb{E}[f(z_{t+1}) - f(z_1)] \leq & \left(G^2 \frac{\beta_1}{1-\beta_1} + 2G^2 \right) \mathbb{E}\left[\sum_{i=2}^{t} \left\| \alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1} \widehat{V}_{i-1}^{-1/2} \right\|_2 \right] \\
& + \left(\frac{3}{2}L + \frac{1}{2} + L^2 \big(\frac{\beta_1}{1-\beta_1}\big)^2 \big(\frac{1}{1-\beta_1}\big)^2 \right) \mathbb{E}\left[\sum_{i=1}^{t} \left\| \alpha_i \widehat{V}_i^{-1/2} g_i \right\|^2 \right] \\
& + \left(1 + L^2 \big(\frac{1}{1-\beta_1}\big)^2 \big(\frac{\beta_1}{1-\beta_1}\big)^2 \right) \big(\frac{\beta_1}{1-\beta_1}\big)^2 G^2 \mathbb{E}\left[\sum_{i=2}^{t-1} \left\| \alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1} \widehat{V}_{i-1}^{-1/2} \right\|_2^2 \right] \\
& + \left(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} \right) (G^2 + D^2) + \Big(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} \Big)^2 D^2 + 2G^2 \mathbb{E}\left[\left\| \alpha_1 V_1^{-1/2} \right\|_2 \right] \\
& - \mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} \nabla f(x_i) \right\rangle \right]
\end{aligned}$$

We define constants C_1, C_2, and C_3 as

$$\begin{aligned}
C_1 &= \frac{3}{2}L + \frac{1}{2} + L^2 \big(\frac{\beta_1}{1-\beta_1}\big)^2 \big(\frac{1}{1-\beta_1}\big)^2 \\
C_2 &= G^2 \frac{\beta_1}{1-\beta_1} + 2G^2 \\
C_3 &= \left(1 + L^2 \big(\frac{1}{1-\beta_1}\big)^2 \big(\frac{\beta_1}{1-\beta_1}\big)^2 \right) \big(\frac{\beta_1}{1-\beta_1}\big)^2 G^2
\end{aligned}$$

By rearranging terms, we obtain

$$\begin{aligned}
\mathbb{E}\left[\sum_{i=1}^{t} \alpha_i \left\langle \nabla f(x_i), V_i^{-1/2} \nabla f(x_i) \right\rangle \right] \leq & \ \mathbb{E}\left[\sum_{i=1}^{t} C_1 \left\| \alpha_i \widehat{V}_i^{-1/2} g_i \right\|^2 + C_2 \sum_{i=2}^{t} \left\| \alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1} \widehat{V}_{i-1}^{-1/2} \right\|_2 \right. \\
& \left. + C_3 \sum_{i=2}^{t-1} \left\| \alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1} \widehat{V}_{i-1}^{-1/2} \right\|_2^2 \right] \\
& + \left(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} \right) (G^2 + D^2) + \Big(\frac{\beta_1}{1-\beta_1} - \frac{\beta_{1,t+1}}{1-\beta_{1,t+1}} \Big)^2 D^2 \\
& + 2G^2 \mathbb{E}\left[\left\| \alpha_1 V_1^{-1/2} \right\|_2 \right] \\
\leq & \ \mathbb{E}\left[\sum_{i=1}^{t} C_1 \left\| \alpha_i \widehat{V}_i^{-1/2} g_i \right\|^2 + C_2 \sum_{i=2}^{t} \left\| \alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1} \widehat{V}_{i-1}^{-1/2} \right\|_2 \right. \\
& \left. + C_3 \sum_{i=2}^{t-1} \left\| \alpha_i \widehat{V}_i^{-1/2} - \alpha_{i-1} \widehat{V}_{i-1}^{-1/2} \right\|_2^2 \right] \\
& + \left(\frac{\beta_1}{1-\beta_1} \right) (G^2 + D^2) + \Big(\frac{\beta_1}{1-\beta_1} \Big)^2 D^2 + 2G^2 \mathbb{E}\left[\left\| \alpha_1 V_1^{-1/2} \right\|_2 \right]
\end{aligned}$$

Finally, we can get

$$\begin{aligned}
& \mathbb{E}\left[\sum_{t=1}^{T} \alpha_i \left\langle \nabla f(x_t), \widehat{V}_t^{-1/2} \nabla f(x_t) \right\rangle \right] \\
\leq & \ \mathbb{E}\left[C_1 \underbrace{\sum_{t=1}^{T} \left\| \alpha_t \widehat{V}_t^{-1/2} g_i \right\|^2}_{\text{Term A}} + C_2 \underbrace{\sum_{t=2}^{T} \left\| \alpha_t \widehat{V}_t^{-1/2} - \alpha_{t-1} \widehat{V}_{t-1}^{-1/2} \right\|_2}_{\text{Term B}} + C_3 \sum_{t=2}^{T-1} \left\| \alpha_t \widehat{V}_t^{-1/2} - \alpha_{t-1} \widehat{V}_{t-1}^{-1/2} \right\|_2^2 \right] + C_4
\end{aligned}$$

with constants

$$C_4 = \left(\frac{\beta_1}{1-\beta_1}\right)(G^2 + D^2) + \left(\frac{\beta_1}{1-\beta_1}\right)^2 D^2 + 2G^2\mathbb{E}\left[\left\|\alpha_1 V_1^{-1/2}\right\|_2\right]$$

with almost same constant for the diagonal version. Lastly, we have $\gamma_t = \lambda_{\min}(\alpha_t \widehat{V}_t^{-1/2})$, so

$$\begin{aligned}\mathbb{E}\left[\sum_{t=1}^{T}\alpha_t\left\langle \nabla f(x_t), \widehat{V}_t^{-1/2}\nabla f(x_t)\right\rangle\right] &\geq \mathbb{E}\left[\sum_{t=1}^{T}\gamma_t\left\|\nabla f(x_t)\right\|^2\right] \\ &\geq \min_{t\in[T]}\mathbb{E}\left[\|\nabla f(x_t)\|^2\right]\sum_{t=1}^{T}\gamma_t\end{aligned}$$

Therefore, we finally have

$$\min_{t\in[T]}\left[\|\nabla f(x_t)\|^2\right] \leq \frac{\mathbb{E}\left[C_1\overbrace{\sum_{t=1}^{T}\left\|\alpha_t\widehat{V}_t^{-1/2}g_t\right\|^2}^{\text{Term A}} + C_2\overbrace{\sum_{t=2}^{T}}^{\text{Term B}}Q_t + C_3\sum_{t=2}^{T-1}Q_t^2\right] + C_4}{\sum_{t=1}^{T}\gamma_t} \triangleq \frac{s_1(T)}{s_2(T)}$$

□

F.3 Proof of Theorem 2

For generalization error bounds, we refer the following references (Hardt et al., 2015; Zheng and Kwok, 2019). Since we have bounded gradient $\|g_t\|_2, \|\nabla f(x)\|_2 \leq G$ and the differentiability of f, we also have G-Lipschitz continuity. Therefore, we obtain the following relation

$$\begin{aligned}\sup_{z}\mathbb{E}_A\left[f(A(S);z) - f(A(S');z)\right] &\leq G\mathbb{E}_A\left[\|A(S) - A(S')\|_2\right] \\ &= G\mathbb{E}_A\left[\|\theta - \theta'\|_2\right]\end{aligned}$$

Therefore, we only have to bound the term $\Delta_t := \|\theta - \theta'\|_2$. From now, we denote $\theta := A(S)$ and $\theta' := A(S')$. We assume $\alpha_t = \alpha$ and $\beta_{1,t} = 0$ for all $t \in [T]$.

$$\begin{aligned}\theta_{T+1} &= \theta_T - \alpha_T(\widehat{V}_T^{1/2} + \delta I)^{-1}m_T \\ &= \cdots \\ &= \theta_1 - \sum_{t=1}^{T}\alpha_t(\widehat{V}_t^{1/2} + \delta I)^{-1}g_t \\ &= \theta_1 - \sum_{t=1}^{T}\alpha_t(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\end{aligned}$$

where z_{i_k} is the selected example at iteration k. Then, we can bound

$$\begin{aligned}\mathbb{E}[\Delta_{T+1}] &= \mathbb{E}\left[\|\theta_{T+1} - \theta'_{T+1}\|_2\right] \\ &= \mathbb{E}\left[\left\|\theta_1 - \sum_{t=1}^{T}\alpha_t(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t}) - \theta'_1 + \sum_{t=1}^{T}\alpha_t(\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta'_t; z'_{i_t})\right\|_2\right] \\ &\leq \mathbb{E}\left[\|\theta_1 - \theta'_1\|_2\right] + \sum_{t=1}^{T}\alpha_t\mathbb{E}\left[\left\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t}) - (\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta'_t; z'_{i_t})\right\|_2\right] \\ &= \sum_{t=1}^{T}\alpha_t\mathbb{E}\left[\left\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t}) - (\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta'_t; z'_{i_t})\right\|_2\right]\end{aligned}$$

The probability of $z_{i_k} = z'_{i_k}$ is $1 - 1/n$. Then,

$$
\begin{aligned}
&\mathbb{E}\left[\left\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t}) - (\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta'_t; z'_{i_t})\right\|_2\right] \\
\leq\ & \frac{1}{n}\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2\big] + \frac{1}{n}\mathbb{E}\big[\|(\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta'_t; z'_{i_t})\|_2\big] \\
&+ \Big(1 - \frac{1}{n}\Big)\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta)^{-1}\nabla f(\theta_t; z_{i_t}) - (\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta'_t; z_{i_t})\|_2\big] \\
\leq\ & \frac{1}{n}\underbrace{\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2\big]}_{T_1} + \frac{1}{n}\underbrace{\mathbb{E}\big[\|(\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta'_t; z'_{i_t})\|_2\big]}_{T_2} \\
&\quad + \Big(1 - \frac{1}{n}\Big)\underbrace{\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t}) - (\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2\big]}_{T_3} \\
&\quad + \Big(1 - \frac{1}{n}\Big)\underbrace{\mathbb{E}\big[\|(\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t}) - (\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta'_t; z_{i_t})\|_2\big]}_{T_4}
\end{aligned}
$$

Let t_0 denote the time when $\widehat{V}_t$ and $\widehat{V}'_t$ becomes full-rank. Then, we can bound T_1 as

$$
\begin{aligned}
&\sum_{t=1}^{T}\alpha_t\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2\big] \\
&\leq \sqrt{T}\sqrt{\sum_{t=1}^{T}\alpha_t^2\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2^2\big]} \\
&= \sqrt{T}\sqrt{\sum_{t=1}^{t_0}\alpha_t^2\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2^2\big] + \sum_{t=t_0+1}^{T}\alpha_t^2\mathbb{E}\big[\mathrm{Tr}\big((\widehat{V}_t^{1/2} + \delta I)^{-2}\nabla f(\theta_t; z_{i_t})\nabla f(\theta_t; z_{i_t})^T\big)\big]} \\
&\leq \sqrt{T}\sqrt{\sum_{t=1}^{t_0}\alpha_t^2\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2^2\big] + \sum_{t=t_0+1}^{T}\alpha_t^2\mathbb{E}\big[\mathrm{Tr}\big(\widehat{V}_t^{-1}\nabla f(\theta_t; z_{i_t})\nabla f(\theta_t; z_{i_t})\big)\big]} \\
&\leq \sqrt{T}\sqrt{\sum_{t=1}^{t_0}\alpha^2\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2^2\big] + \frac{\alpha^2}{1-\beta_2}\mathbb{E}\big[\log\det(\widehat{V}_T) - \log\det(\widehat{V}_{t_0}) + d(T - t_0)\log\frac{1}{\beta_2}\big]} \\
&= \frac{\alpha\sqrt{T}}{\sqrt{1-\beta_2}}\sqrt{(1-\beta_2)\sum_{t=1}^{t_0}\mathbb{E}\big[\|(\widehat{V}_t^{1/2} + \delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2^2\big] + \mathbb{E}\big[\log\det(\widehat{V}_T) - \log\det(\widehat{V}_{t_0})\big] + d(T - t_0)\log\frac{1}{\beta_2}} \\
&\leq \frac{\alpha\sqrt{T}}{\sqrt{1-\beta_2}}\sqrt{(1-\beta_2)\sum_{t=1}^{t_0}\mathbb{E}\big[\|(\delta I)^{-1}\nabla f(\theta_t; z_{i_t})\|_2^2\big] + \mathbb{E}\big[\log\det(\widehat{V}_T) - \log\det(\widehat{V}_{t_0})\big] + d(T - t_0)\log\frac{1}{\beta_2}} \\
&\leq \frac{\alpha\sqrt{T}}{\sqrt{1-\beta_2}}\sqrt{\frac{t_0(1-\beta_2)G^2}{\delta^2} + \mathbb{E}\left[\log\frac{|\widehat{V}_T|}{|\widehat{V}_{t_0}|}\right] + d(T - t_0)\log\frac{1}{\beta_2}}
\end{aligned}
$$

In the same way, we can bound T_2 as

$$
\sum_{t=1}^{T}\alpha_t\mathbb{E}\big[\|(\widehat{V}_t'^{1/2} + \delta I)^{-1}\nabla f(\theta'_t; z'_{i_t})\|_2\big] \leq \frac{\alpha\sqrt{T}}{\sqrt{1-\beta_2}}\sqrt{\frac{t_0(1-\beta_2)G^2}{\delta^2} + \mathbb{E}\left[\log\frac{|\widehat{V}'_T|}{|\widehat{V}'_{t_0}|}\right] + d(T - t_0)\log\frac{1}{\beta_2}}
$$

For notational convenience, we set the function g as

$$
g(\widehat{V}_T) = \frac{\alpha\sqrt{T}}{\sqrt{1-\beta_2}}\sqrt{\frac{t_0(1-\beta_2)G^2}{\delta^2} + \mathbb{E}\left[\log\frac{|\widehat{V}_T|}{|\widehat{V}_{t_0}|}\right] + d(T - t_0)\log\frac{1}{\beta_2}}
$$

Now, we can easily bound T_3 and T_4 as

$$\begin{aligned}\mathbb{E}\big[\|(\widehat{V}_t^{1/2}+\delta I)^{-1}\nabla f(\theta_t;z_{i_t})-(\widehat{V}_t'^{1/2}+\delta I)^{-1}\nabla f(\theta_t;z_{i_t})\|_2\big] &\le G\mathbb{E}\big[|\!|\!|(\widehat{V}_t^{1/2}+\delta I)^{-1}-(\widehat{V}_t'^{1/2}+\delta I)^{-1}|\!|\!|_2\big]\\ &\le G\mathbb{E}\big[|\!|\!|(\widehat{V}_t^{1/2}+\delta I)^{-1}|\!|\!|_2+|\!|\!|(\widehat{V}_t'^{1/2}+\delta I)^{-1}|\!|\!|_2\big]\end{aligned}$$

and

$$\mathbb{E}\big[\|(\widehat{V}_t'^{1/2}+\delta I)^{-1}\nabla f(\theta_t;z_{i_t})-(\widehat{V}_t'^{1/2}+\delta I)^{-1}\nabla f(\theta_t';z_{i_t})\|_2\big]\le L\mathbb{E}\big[|\!|\!|(\widehat{V}_t'^{1/2}+\delta I)^{-1}|\!|\!|_2\Delta_t\big]$$

from $\|Ax\|_2\le |\!|\!|A|\!|\!|_2\|x\|_2$ and Lipschitz continuous gradients. Combining all the terms, we finally have

$$\mathbb{E}\big[\Delta_{T+1}\big]\le\frac{\alpha\sqrt{T}}{n\sqrt{1-\beta_2}}\left[\sqrt{g(\widehat{V}_T)}+\sqrt{g(\widehat{V}_T')}\right]+\alpha\Big(1-\frac{1}{n}\Big)J_T$$

where

$$g(\widehat{V}_T)=\frac{t_0(1-\beta_2)G^2}{\delta^2}+\mathbb{E}\big[\underbrace{\log|\widehat{V}_T|-\log|\widehat{V}_{t_0}|}_{\text{Term C}}\big]+d(T-t_0)\log\frac{1}{\beta_2}$$

and

$$J_T=G\sum_{t=1}^{T}\mathbb{E}\big[\underbrace{|\!|\!|(\widehat{V}_t^{1/2}+\delta I)^{-1}|\!|\!|_2}_{\text{Term D}}+\underbrace{|\!|\!|(\widehat{V}_t'^{1/2}+\delta I)^{-1}|\!|\!|_2}_{\text{Term D}}\big]+L\sum_{t=1}^{T}\mathbb{E}\big[\underbrace{|\!|\!|(\widehat{V}_t'^{1/2}+\delta I)^{-1}|\!|\!|_2}_{\text{Term D}}\Delta_t\big]$$

F.4 Generalization Bounds for AdaGrad

The main difference with EMA-BASED algorithms is the way of bounding T_1. For ADAGRAD, we set $\alpha_t=\alpha/\sqrt{t}$ and $\widehat{V}_t=\frac{1}{t}\sum_{i=1}^{t}g_ig_i^T=:\frac{1}{t}G_t$ as in Corollary 1.

Let t_0 denote the time when $\widehat{V}_t$ and $\widehat{V}_t'$ becomes full-rank. Then, we can bound T_1 for ADAGRAD as

$$\begin{aligned}&\sum_{t=1}^{T}\alpha_t\mathbb{E}\big[\|(\widehat{V}_t^{1/2}+\delta I)^{-1}\nabla f(\theta_t;z_{i_t})\|_2\big]\\ &\le\sqrt{T}\sqrt{\sum_{t=1}^{T}\alpha_t^2\mathbb{E}\big[\|(\widehat{V}_t^{1/2}+\delta I)^{-1}\nabla f(\theta_t;z_{i_t})\|_2^2\big]}\\ &=\sqrt{T}\sqrt{\sum_{t=1}^{t_0}\alpha_t^2\mathbb{E}\big[\|(\widehat{V}_t^{1/2}+\delta I)^{-1}\nabla f(\theta_t;z_{i_t})\|_2^2\big]+\sum_{t=t_0+1}^{T}\alpha_t^2\mathbb{E}\big[\mathrm{Tr}\big((\widehat{V}_t^{1/2}+\delta I)^{-2}\nabla f(\theta_t;z_{i_t})\nabla f(\theta_t;z_{i_t})^T\big)\big]}\\ &\le\sqrt{T}\sqrt{\sum_{t=1}^{t_0}\alpha_t^2\mathbb{E}\big[\|(\widehat{V}_t^{1/2}+\delta I)^{-1}\nabla f(\theta_t;z_{i_t})\|_2^2\big]+\sum_{t=t_0+1}^{T}\alpha_t^2\mathbb{E}\big[\mathrm{Tr}\big(\widehat{V}_t^{-1}\nabla f(\theta_t;z_{i_t})\nabla f(\theta_t;z_{i_t})\big)\big]}\\ &\le\sqrt{T}\sqrt{\sum_{t=1}^{t_0}\alpha^2\mathbb{E}\big[\|(\widehat{V}_t^{1/2}+\delta I)^{-1}\nabla f(\theta_t;z_{i_t})\|_2^2\big]+\alpha^2\mathbb{E}\big[\log|\widehat{V}_T|-\log|\widehat{V}_{t_0}|+d\log\frac{T}{t_0}\big]}\\ &\le\alpha\sqrt{T}\sqrt{\sum_{t=1}^{t_0}\mathbb{E}\big[\|(\delta I)^{-1}\nabla f(\theta_t;z_{i_t})\|_2^2\big]+\mathbb{E}\big[\log|\widehat{V}_T|-\log|\widehat{V}_{t_0}|+d\log\frac{T}{t_0}\big]}\\ &\le\alpha\sqrt{T}\sqrt{\frac{t_0G^2}{\delta^2}+\mathbb{E}\big[\log|\widehat{V}_T|-\log|\widehat{V}_{t_0}|+d\log\frac{T}{t_0}\big]}\end{aligned}$$

We can similarly bound the term T_2 only replacing $\widehat{V}_t$ with $\widehat{V}_t'$. Also, the remaining terms T_3 and T_4 can be bound in the same way as in section F.3. Therefore, we have

$$\mathbb{E}\big[\Delta_{T+1}\big] \leq \frac{\alpha\sqrt{T}}{n}\left[\sqrt{g(\widehat{V}_T)} + \sqrt{g(\widehat{V}_T')}\right] + \alpha\Big(1 - \frac{1}{n}\Big) J_T$$

where

$$g(\widehat{V}_T) = \frac{t_0 G^2}{\delta^2} + \mathbb{E}\big[\underbrace{\log|\widehat{V}_T| - \log|\widehat{V}_{t_0}|}_{\text{Term C}}\big] + d\log\frac{T}{t_0}$$

and

$$J_T = G\sum_{t=1}^{T}\mathbb{E}\big[\underbrace{\big\vert\!\big\vert\!\big\vert(\widehat{V}_t^{1/2} + \delta I)^{-1}\big\vert\!\big\vert\!\big\vert_2}_{\text{Term D}} + \underbrace{\big\vert\!\big\vert\!\big\vert(\widehat{V}_t'^{1/2} + \delta I)^{-1}\big\vert\!\big\vert\!\big\vert_2}_{\text{Term D}}\big] + L\sum_{t=1}^{T}\mathbb{E}\big[\underbrace{\vert\!\vert\!\vert(\widehat{V}_t'^{1/2} + \delta I)^{-1}\vert\!\vert\!\vert_2}_{\text{Term D}}\Delta_t\big]$$