NAMSG: AN EFFICIENT METHOD FOR TRAINING NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce NAMSG, an adaptive first-order algorithm for training neural networks. The method is efficient in computation and memory, and is straightforward to implement. It computes the gradients at configurable remote observation points, in order to expedite the convergence by adjusting the step size for directions with different curvatures in the stochastic setting. It also scales the updating vector elementwise by a nonincreasing preconditioner to take the advantages of AMSGRAD. We analyze the convergence properties for both convex and nonconvex problems by modeling the training process as a dynamic system, and provide a strategy to select the observation factor without grid search. A data-dependent regret bound is proposed to guarantee the convergence in the convex setting. The method can further achieve a $O(log(T))$ regret bound for strongly convex functions. Experiments demonstrate that NAMSG works well in practical problems and compares favorably to popular adaptive methods, such as ADAM, NADAM, and AMSGRAD.

1 INTRODUCTION AND RELATED WORK

Training deep neural networks (Collobert et al., 2011; Hinton et al., 2012; Amodei et al., 2016; He et al., 2016) with large datasets costs a huge amount of time and computational resources. Efficient optimization methods are urgently required to accelerate the training process.

First-order optimization methods (Robbins & Monro, 1951; Polyak, 1964; Bottou, 2010; Sutskever et al., 2013; Kingma & Ba, 2015; Bottou et al., 2018) are currently the most popular for training neural networks. They are easy to implement since only first-order gradients are introduced as input. Besides, they require low computation overheads except for computing gradients, which is of the same computational complexity as just evaluating the function. Compared with second-order methods (Nocedal, 1980; Martens, 2010; Byrd et al., 2016), they are more effective to handle gradient noise.

Sutskever et al. (2013) show that the momentum is crucial to improve the performance of SGD. Momentum methods, such as HB Polyak (1964), can amplify steps in low-curvature eigen-directions of the Hessian through accumulation, although careful tuning is required to ensure fine convergence along the high-curvature directions. Sutskever et al. (2013) also rewrite the Nesterov's Accelerated Gradient (NAG) (Nesterov, 1983) in a momentum form, and show the performance improvement over HB. The method computes the gradient at a observation point ahead of the current point along the last updating direction. They illustrate that NAG suppresses the step along high curvature eigen-directions in order to prevent oscillations. However, all these approaches are approximation of their original forms derived for exact gradients, without fully study on gradient noise. Kidambi et al. (2018) show the insufficiency of HB and NAG in stochastic optimization, especially for small minibatches. They further present ASGD (Jain et al., 2018) and show significant improvements. However, the method requires tuning of 3 parameters, leading to huge costs that impedes its practical applications.

Among variants of SGD methods, adaptive methods that scale the gradient elementwise by some form of averaging of the past gradients are particularly successful. ADAGRAD (Duchi et al., 2011) is the first popular method in this line. It is well-suited for sparse gradients since it uses all the past gradients to scale the update. Nevertheless, it suffers from rapid decay of step sizes, in cases of nonconvex loss functions or dense gradients. Subsequent adaptive methods, such as RMSPROP (Tieleman & Hinton., 2012), ADADELTA (Zeiler, 2012), ADAM (Kingma & Ba, 2015), and NADAM (Dozat, 2016), mitigate this problem by using the exponential moving averages of squared past gradients.

However, Reddi et al. (2018) show that ADAM does not converge to optimal solutions in some convex problems, and the analysis extends to RMSPROP, ADADELTA, and NADAM. They propose AMSGRAD, which fixes the problem and shows improvements in experiments.

In this paper, we propose NAMSG, that is an efficient first-order method for training neural networks. The name is derived from combining a configurable NAG method (CNAG) and AMSGRAD. NAMSG computes the stochastic gradients at configurable observation points ahead of the current parameters along the last updating direction. Nevertheless, instead of approximating NAG for exact gradients, it adjusts the learning rates for eigen-directions with different curvatures to expedite convergence in the stochastic setting, by selecting the observation distance. It also scales the update vector elementwisely using the nonincreasing preconditioner of AMSGRAD. We analyze the convergence properties by modeling the training process as a dynamic system, reveal the benefits of remote gradient observations and provide a strategy to select the observation factor without grid search. A regret bound is introduced in the convex setting, and it is further improved for strongly convex functions. Finally, we present experiments to demonstrate the efficiency of NAMSG in real problems.

2 THE NAMSG SCHEME

Before further description, we introduce the notations following Reddi et al. (2018), with slight abuse of notation. The letter t denotes iteration number, d denotes the dimension of vectors and matrices, ϵ denotes a predefined positive small value, and S_+^d denotes the set of all positive definite $d \times d$ matrix. For a vector $a \in R_d$ and a matrices $M \in R_d \times R_d$, we use a/M to denote $M^{-1}a$, $\mathrm{diag}(a)$ to denote a square diagonal matrix with the elements of a on the main diagonal, M_i to denote the i^{th} row of M, and $\sqrt{M}$ to denote $M^{1/2}$. For any vectors $a, b \in R_d$, we use $\sqrt{a}$ for elementwise square root, a^2 for elementwise square, a/b for elementwise division, and $\max(a, b)$ to denote elementwise maximum. For any vector $\theta_i \in R_d$, $\theta_{i,j}$ denotes its j^{th} coordinate where $j \in \{1, 2, \ldots, d\}$. We define $\mathcal{F} \subset R_d$ as the feasible set of points. Assume that $\mathcal{F}$ has bounded diameter D_∞, i.e. $\|x - y\| \leq D_\infty$ for any $x, y \in \mathcal{F}$, and $\|\nabla f_t(x)\|_\infty \leq G_\infty$, $\|\nabla f_t(x)\|_1 \leq G_1$ for all $x \in \mathcal{F}$. The projection operation is defined as $\Pi_{\mathcal{F},A}(y) = \arg\min_{x \in \mathcal{F}} \|A^{1/2}(x - y)\|$ for $A \in S_+^d$ and $y \in R_d$.

In the context of machine learning, we consider the minimization problem of a stochastic function,

$$\min_{x \in R^d} F(x) = \mathbb{E}_\xi[f(x, \xi)], \tag{1}$$

where x is a d dimensional vector consisting of the parameters of the model, and ξ is a random datum consisting of an input-output pair. Since the distribution of ξ is generally unavailable, the optimizing problem (1) is approximated by minimizing the empirical risk on the training set $\{\zeta_1, \zeta_2, ..., \zeta_N\}$, as

$$\min_{x \in R^d} \mathcal{L}(x) = \frac{1}{N} \sum_{i=1}^{N} f(x, \zeta_i). \tag{2}$$

In order to save computation and avoid overfitting, it is common to estimate the objective function and its gradient with a minibatch of training data, as

$$f_t(x) = \frac{1}{b} \sum_{i \in S_t} f(x, \zeta_i), \quad \text{and} \quad \nabla f_t(x) = \frac{1}{b} \sum_{i \in S_t} \nabla f(x, \zeta_i), \tag{3}$$

where the minibatch $S_t \subset \{1, 2, ..., N\}$, and $b = |S_t|$ is the size of S_t.

Firstly, we propose a configurable NAG method (CNAG). Since the updating directions are partially maintained in momentum methods, gradients computed at observation points, which lie ahead of the current point along the last updating direction, contain the predictive information of the forthcoming update. The remote observation points are defined as $\dot{x}_t = x_t - \eta_t u_{t-1}$ where u_{t-1} is the updating vector, and $\dot{x}_1 = x_1$.

By computing the gradient at a configurable observation point $\dot{x}_t$, and substituting the gradient with the observation gradient in the HB update, we obtain the original form of CNAG, as

$$\begin{aligned} m_t &= \beta_t m_{t-1} + (1 - \beta_t) \nabla f_t(\dot{x}_t) \\ x_{t+1} &= x_t - \alpha_t m_t \\ \dot{x}_{t+1} &= x_t - (1 + \eta_t) \alpha_t m_t, \end{aligned} \tag{4}$$

Algorithm 1 NAMSG Algorithm

Input: initial parameter vector x_1, coefficients $\{\alpha_t\}_{t=1}^T$, $\{\beta_{1t}\}_{t=1}^T$, β_2, ϵ, iteration number T
Output: parameter vector x_T

Set $m_0 = 0$, $v_0 = 0$, and $\hat{v}_0 = \epsilon$.
for $t = 1$ to $T - 1$ **do**
  $g_t = \nabla f_t(x_t)$.
  $m_t = \beta_{1t} m_{t-1} + (1 - \beta_{1t}) g_t$.
  $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$.
  $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$, $\hat{V}_t = \mathrm{diag}(v_t)$.
  $\hat{m}_t = (1 - \mu_t) m_t + \mu_t g_t$, where $\mu_t = \eta_t(1 - \beta_{1t})/\beta_{1t}$.
  $x_{t+1} = \Pi_{\mathcal{F}, \sqrt{\hat{V}_t}}(x_t - \alpha_t \hat{m}_t / \sqrt{\hat{v}_t})$.
end for

where α_t, β_t, η_t are configurable coefficients, and $m_0 = 0$. The observation distance η_t can be configured to accommodate gradient noise, instead of $\eta_t = \beta_t$ in NAG (Sutskever et al., 2013).

Both x_t and $\dot{x}_t$ are required to update in (4). To make the method more efficient, we simplify the update by approximation. Assume that the coefficients α_t, β_{1t}, and η_t, change very slowly between adjacent iterations. Substituting x_t by $\dot{x}_t + \eta_{t-1}\alpha_{t-1} m_{t-1}$, we obtain the concise form of CNAG, as

$$\begin{aligned} m_t &= \beta_t m_{t-1} + (1 - \beta_t) \nabla f_t(x_t) \\ x_{t+1} &= x_t - \alpha_t \left((1 - \mu_t) m_t + \mu_t \nabla f_t(x_t) \right), \end{aligned} \tag{5}$$

where the observation factor $\mu_t = \eta_t(1 - \beta_t)/\beta_t$, and we use x instead of $\dot{x}$ for simplicity.

In practical computation of CNAG, we further rearrange the update form as

$$\begin{aligned} \tilde{m}_t &= \frac{\alpha_t (1 - \mu_t)}{\alpha_{t-1} (1 - \mu_{t-1})} \beta_t \tilde{m}_{t-1} + \alpha_t (1 - \beta_t) (1 - \mu_t) \nabla f_t(x_t) \\ x_{t+1} &= x_t - \tilde{m}_t - \alpha_t \mu_t \nabla f_t(x_t), \end{aligned} \tag{6}$$

where only 3 scalar vector multiplications and 3 vector additions are required per iteration besides the gradient computation. Hereinafter, we still use (5) for simplicity in expressions.

Then, we study the relation of CNAG and ASGD, that guides the selection of the momentum coefficient. Jain et al. (2018) shows that ASGD improves on SGD in any information-theoretically admissible regime. By taking a long step as well as short step and an appropriate average of both of them, ASGD tries to make similar progress on different eigen-directions. It takes 3 hyper-parameters: short step $\ddot{\alpha}$, long step parameter $\ddot{\kappa}$, and statistical advantage parameter $\ddot{\xi}$. $\ddot{\alpha}$ is the same as the step size in SGD. For convex functions, $\ddot{\kappa}$ is an estimation of the condition number. The statistical advantage parameter $\ddot{\xi} \leq \sqrt{\ddot{\kappa}}$ captures trade off between statistical and computational condition numbers, and $\ddot{\xi} \ll \sqrt{\ddot{\kappa}}$ in high stochasticity regimes. These hyper-parameters vary in large ranges, and are difficult to estimate. The huge costs in tuning limits the application of ASGD.

The appendix shows that CNAG is a more efficient equivalent form of ASGD. For CNAG with constant hyper-parameters, the momentum coefficient $\beta_t = \beta = (\ddot{\kappa} - 0.49\ddot{\xi})/(\ddot{\kappa} + 0.7\ddot{\xi})$. Since the condition number is generally large in real high dimension problems, and the statistical advantage parameter $\ddot{\xi} \leq \sqrt{\ddot{\kappa}}$, β is close to 1. To sum up, the equivalence of CNAG and ASGD shows that in order to narrow the gap between the step sizes on eigen-directions with different curvatures, the momentum coefficient β should be close to 1.

Finally, we form NAMSG by equipping CNAG with the nonincreasing preconditioner of AMSGRAD, and project the parameter vector x into the feasible set $\mathcal{F}$. Algorithm 1 shows the pseudo code of NAMSG. Compared with AMSGRAD, NAMSG requires low computation overheads, as a scalar vector multiplication and a vector addiction per iteration, which are much cheaper than the gradient estimation. Almost no more memory is needed if the vector operations are run by pipelines. In most cases, especially when weight decay is applied for regularization, which limits the norm of the parameter vectors, the projection can also be omitted in implementation to save computation.

3 AN ANALYSIS ON THE EFFECT OF REMOTE GRADIENT OBSERVATIONS

In Algorithm 1, the observation factor μ_t is configurable to accelerate convergence. However, it is costly to select it by grid search. In this section we analyze the convergence rate in a local stochastic quadratic optimization setting by investigating the optimizing process as a dynamic system, and reveal the effect of remote gradient observation for both convex and non-convex problems. Based on the analysis, we provide the default values and a practical strategy to set the observation factor without grid search.

The problem (1) can be approximated locally as a stochastic quadratic optimization problem, as

$$\min_{x\in\hat{\mathcal{F}}} \hat{\Phi}(x) = \frac{1}{2}(x-x^*)^T\hat{H}(x-x^*), \tag{7}$$

where $\hat{\mathcal{F}}$ is a local set of feasible parameter points. In the problem, the gradient observation is noisy as $\nabla f_t(x) = \nabla\hat{\Phi}(x) + \acute{g}_t$, where $\acute{g}_t$ is the gradient noise.

Consider the optimization process of NAMSG, and ignore the projections for simplicity. Since $\hat{v}_t$ varies slowly when t is large, we can ignore the change of $\hat{v}_t$ between recent iterations. The operation of dividing the update by $\sqrt{\hat{v}_t}$ can be approximated by solving a preconditioned problem, as

$$\min_{\check{x}\in\check{\mathcal{F}}} \check{\Phi}(\check{x}) = \frac{1}{2}(\check{x}-\check{x}^*)^T\hat{V}_t^{-1/4}\hat{H}\hat{V}_t^{-1/4}(\check{x}-\check{x}^*), \tag{8}$$

where $\check{x} = \hat{V}_t^{1/4}x$, $\check{x}^* = \hat{V}_t^{1/4}x^*$. Define the preconditioned Hessian $H = \hat{V}_t^{-1/4}\hat{H}\hat{V}_t^{-1/4}$, which is supposed to have improved condition number compared with $\hat{H}$, in the convex setting.

Then, we model the optimization process as a dynamic system. Solving the quadratic problem (7) by NAMSG is equal to solving the preconditioned problem (8) by CNAG, as

$$\begin{aligned} \check{m}_t &= \beta\check{m}_{t-1} + (1-\beta)\nabla f_t(\check{x}_t) \\ \check{x}_{t+1} &= \check{x}_t - \alpha\left((1-\mu)\check{m}_t + \mu\nabla f_t(\check{x}_t)\right), \end{aligned} \tag{9}$$

where the preconditioned stochastic function $\check{f}_t(\check{x}) = f_t(\hat{V}_t^{-1/4}\check{x})$, the initial momentum $\check{m}_0 = 0$, the coefficients $\alpha = (1-\beta_{1t})\alpha_t$, $\beta = \beta_{1t}$, and $\eta = \eta_t$ are considered as constants.

We use ν to denote a unit eigenvector of the Hessian H, and the corresponding eigenvalue is λ. We define the coefficients as $\dot{s}_t = \langle\nu, \check{x}_t\rangle$, $\dot{v}_t = \langle\nu, \check{m}_t\rangle$. According to (9), the coefficients are updated as

$$\begin{cases} \dot{v}_{t+1} = \beta\dot{v}_t + (1-\beta)\lambda(\dot{s}_t + \delta_t), \\ \dot{s}_{t+1} = \dot{s}_t - \alpha\beta(1-\mu)\dot{v}_t - \alpha(1-\beta(1-\mu))\lambda(\dot{s}_t + \delta_t) \end{cases} \tag{10}$$

where the gradient error coefficient $\delta_t = \langle\hat{V}_t^{-1/4}\acute{g}_t, \nu\rangle/\lambda$.

Substituting $\dot{v}_t$ by $\tilde{v}_t = \alpha\dot{v}_t$, and denote $\tau = \alpha\lambda$, we rewrite the update (10) into a dynamic systemp as

$$\begin{bmatrix}\tilde{v}_{t+1}\\ \dot{s}_{t+1}\end{bmatrix} = A\begin{bmatrix}\tilde{v}_t\\ \dot{s}_t\end{bmatrix} + b\delta_t, A = \begin{bmatrix}\beta & (1-\beta)\tau\\ -\beta(1-\mu) & 1-(1-\beta(1-\mu))\tau\end{bmatrix}, b = \begin{bmatrix}(1-\beta)\tau\\ -(1-\beta(1-\mu))\tau\end{bmatrix}, \tag{11}$$

where A is the gain matrix. The eigenvalues of A are

$$r_1 = \frac{1}{2}\left(\rho - \sqrt{\rho^2 - 4\beta(1-\mu\tau)}\right), \quad r_2 = \frac{1}{2}\left(\rho + \sqrt{\rho^2 - 4\beta(1-\mu\tau)}\right), \tag{12}$$

where $\rho = 1 + \beta - \tau(1-\beta(1-\mu))$. Denote the corresponding unit eigenvectors as w_1 and w_2, that are solved numerically since the expressions are too complicated.

Define the coefficients c_1, c_2, d_1, d_2 satisfying

$$c_1w_1 + c_2w_2 = b, d_1w_1 + d_2w_2 = \begin{bmatrix}\tilde{v}_1\\ \dot{s}_1\end{bmatrix}. \tag{13}$$

From (11), (12) and (13), we obtain

$$\begin{bmatrix}\tilde{v}_{t+1}\\ \dot{s}_{t+1}\end{bmatrix} = r_1^t d_1 w_1 + r_2^t d_2 w_2 + \sum_{l=1}^{t}\delta_l\left(r_1^{t-l}c_1w_1 + r_2^{t-l}c_2w_2\right). \tag{14}$$

Assume that $\delta_t = \sigma\delta$, where δ obeys the standard normal distribution, and σ is the standard deviation of δ_t. From (14), we obtain $\mathbb{E}(\dot{s}_{t+1}) = r_1^t d_1 w_{1,2} + r_2^t d_2 w_{2,2}$ and

$$\lim_{t\to+\infty} \mathrm{Var}(\dot{s}_t) = \left(|c_1|^2|w_{1,2}|^2/(1-|r_1|^2) + |c_2|^2|w_{2,2}|^2/(1-|r_2|^2)\right.$$
$$\left. +2\,\mathrm{Re}\left(\overline{c_1}c_2\overline{w_{1,2}}w_{2,2}/(1-\overline{r_1}r_2)\right)\right)\delta^2, \text{ if } \max(|r_1|,|r_2|) < 1. \tag{15}$$

According to the analysis in Section 2, we recommend the momentum factor $\beta = 0.999$. Figure 1 presents the gain factor $g_{fac} = \max(|r_1|, |r_2|)$ and the stand deviation limit $\lim_{t\to+\infty} \mathrm{Std}(\dot{s}_t)$ of CNAG. It is shown that compared with HB ($\mu = 0$), a proper observation factor μ improves the convergence rate significantly, and also accelerates the divergence in nonconvex problems where $\tau = \alpha\lambda < 0$. When the step size α is constant, compared with large curvatures, a small curvature λ converges much slower, forming the bottleneck of the whole training process. The problem can be alleviated by using a large mu. However, the noise level also increases along with mu when α is constant, that prohibits too large μ. Consequently, we recommend $mu = 0.1$ to achieve fast convergence speed, and $mu = 0.2$ to improve generalization at the cost of more iterations, since higher noise level is beneficial for expanding the range of exploration. Only the step size α is left for grid search.

Figure 1 also shows that a large β and a proper μ ensures a large convergence domain, while $0 < \tau < 2$ is required for convergence in SGD ($\beta = 0$). Since the range of eigenvalue λ is problem-dependent, a large maximum τ (denoted by τ_{max}) allows large step sizes. As shown in Figure 1 (a), μ does not effect g_{fac} significantly for a tiny range of τ close to 0. Then, g_{fac} decreases almost linearly according to τ to the minimum. Consequently, training with a large step size α and small μ is beneficial for both the convergence of tiny positive λ, and the divergence of tiny negative λ in nonconvex settings. While selecting a smaller μ and scaling α proportional to $\mathrm{argmin}_\mu g_{\text{fac}}$, the λ to minimize g_{fac} is unchanged, and the convergence rate for $0 < \lambda < \tau_{max}/\alpha$ is generally improved according to Figure 1. However, the noise level also increases, that prohibits too large α. We propose a hyper-parameter policy named observation boost (OBSB). The policy performs grid search for a small portion of iterations using a small μ to select an optimal initial α. In training, when the loss flattens, it doubles μ, and scalesp α proportional to $\mathrm{argmin}_\mu g_{\text{fac}}$. The recommend initial μ is 0.05.

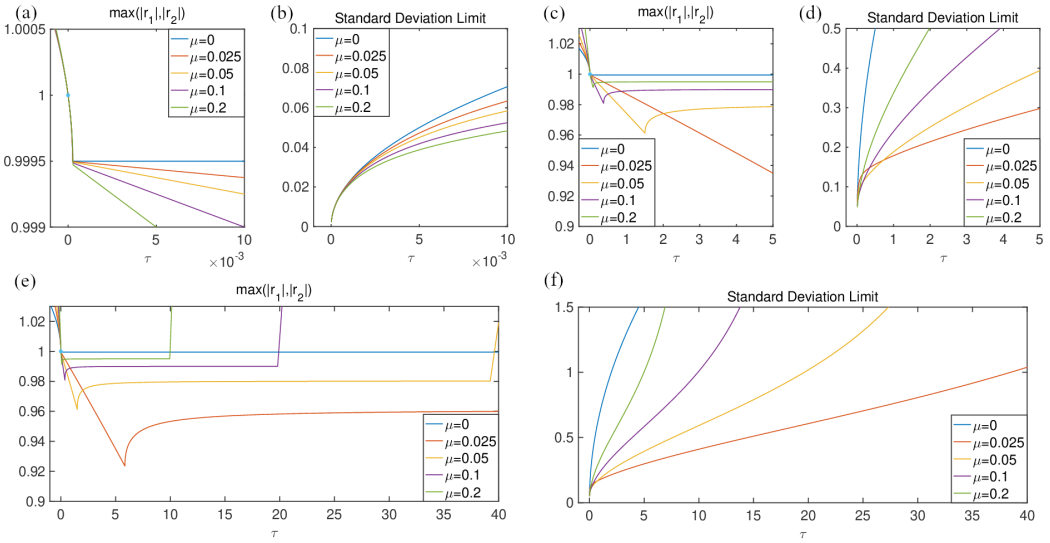

Figure 1: The gain factor $\max(|r_1|, |r_2|)$ and stand deviation limit of CNAG when $\beta = 0.999$: (a) and (b) for $\tau \in [-0.001, 0.01]$, (c) and (d) $\tau \in [-0.3, 5]$, (e) and (f) for $\tau \in [-1, 40]$.

4 Convergence Analysis

In this section, we provide a data dependent regret bound of NAMSG in the convex setting, and further improve the bound for strongly convex functions.

Since the sequence of cost functions $f_t(x)$ are stochastic, we evaluate the convergence property of our algorithm by regret, which is the sum of all the previous difference between the online prediction $f_t(x_t)$ and the best fixed point parameter $f_t(x^*)$ for all the previous steps, defined as $R_T = \sum_{t=1}^{T}(f_t(x_t) - f_t(x^*))$. When the regret of an algorithm satisfies $R_T = o(T)$, the algorithm converges to the optimal parameters on average.

Assuming that $\alpha_t \geq \alpha_{t+1}$, NAMSG insures $\Gamma_t = \sqrt{\hat{V}_{t+1}}/\alpha_{t+1} - \sqrt{\hat{V}_t}/\alpha_t \in S_+^d$. The positive definiteness of Γ_t results in a nonincreasing step size and avoids the non-convergence of ADAM. Following Reddi et al. (2018), we derive the following key results for NAMSG.

Theorem 1. Let $\{x_t\}$, $\{\hat{v}_t\}$ and $\{v_t\}$ be the sequences obtained from Algorithm 1, $\alpha_t = \alpha/\sqrt{t}$, $\beta_1 = \beta_{11} < 1$, $0 \leq \beta_{1t+1} \leq \beta_{1t}$, $\gamma = \beta_1/\sqrt{\beta_2} < 1$, $1 - \beta_1 \leq \mu_t = \mu < 1$, for all $t \in \{1, \cdots, T\}$, and $x \in \mathcal{F}$. We have the following bound on the regret

$$R_T \leq \frac{1}{(1-\beta_1(1-\mu))}\left(\frac{D_\infty^2\sqrt{T}}{2\alpha}\sum_{i=1}^{d}\hat{v}_{T,i}^{1/2} + \frac{(1-\mu)D_\infty^2}{2}\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_{1t}\hat{v}_{t,i}^{1/2}}{\alpha_t}\right.$$
$$\left. + \left(\frac{3\beta_1^2}{2(1-\beta_1)(1-\gamma)} + \mu^2\right)\frac{\alpha\sqrt{1+\log(T)}}{\sqrt{1-\beta_2}}\sum_{i=1}^{d}\|g_{1:T,i}\|_2\right). \tag{16}$$

By compared with the regret bound of AMSGRAD (Reddi et al., 2018), we find that the regret bounds of the two methods have the similar form. However, when β_1 and γ are close to 1, which is the typical situation, NAMSG has lower coefficients on all of the 3 terms.

From Theorem 1, we can immediately obtain the following corollary.

Corollary 1. Suppose $\beta_{1t} = \beta_1/t$, then we have

$$R_T \leq \frac{D_\infty^2\sqrt{T}(1+2(1-\mu)\beta_1)}{2\alpha(1-\beta_1(1-\mu))}\sum_{i=1}^{d}\hat{v}_{T,i}^{1/2} + \frac{\left(3\beta_1^2+2(1-\beta_1)(1-\gamma)\mu^2\right)\alpha\sqrt{1+\log(T)}}{2(1-\beta_1)(1-\beta_1(1-\mu))(1-\gamma)\sqrt{1-\beta_2}}\sum_{i=1}^{d}\|g_{1:T,i}\|_2. \tag{17}$$

The bound in Corollary 1 is considerably better than $O(\sqrt{dT})$ regret of SGD when $\sum_{i=1}^{d}\hat{v}_{t,i}^{1/2} \ll \sqrt{d}$ and $\sum_{i=1}^{d}\|g_{1:T,i}\| \ll \sqrt{dT}$ (Duchi et al., 2011).

For strongly convex functions, NAMSG further achieves a $O(\log(T))$ regret bound with $O(1/t)$ step size (Bottou et al., 2018; Wang et al., 2019) under certain assumptions.

Theorem 2. Assume that $\forall x_1, x_2 \in \mathcal{F}$, $f_t(x_1) \geq f_t(x_2) + \nabla f_t(x_2)^\top(x_1 - x_2) + \frac{\lambda}{2}\|x_1 - x_2\|^2$, where λ is a positive constant. Let $\{x_t\}$, $\{\hat{v}_t\}$ and $\{v_t\}$ be the sequences obtained from Algorithm 1. The initial step size $\alpha \geq \max_{i\in\{1,\cdots,d\}}\left(t\hat{v}_{t,i}^{1/2} - (t-1,i)\hat{v}_{t-1}^{1/2}\right)/\left((1-\beta_1(1-\mu))\lambda\right)$, $\alpha_t = \alpha/t$, $0 \leq \beta_t = \beta_1/t^2 < 1$, $\gamma = \beta_1/\sqrt{\beta_2} < 1$, $1 - \beta_1 \leq \mu_t = \mu < 1$, $\epsilon \to 0^+$, for all $t \in \{1, \cdots, T\}$, and $x \in \mathcal{F}$. We have the following bound on the regret

$$R_T \leq \left(\frac{\alpha G_1}{\sqrt{1-\beta_2}}\left(\frac{3}{2}\frac{\beta_1^2}{(1-\beta_1)(1-\gamma)} + \mu^2\right) + \frac{(1-\mu)\beta_1 D_\infty^2}{2\alpha}\sum_{i=1}^{d}\hat{v}_{T,i}^{1/2}\right)\frac{1+\log(T)}{1-\beta_1(1-\mu)}. \tag{18}$$

When the gradients are sparse, satisfying $\sum_{i=1}^{d}\hat{v}_{T,i}^{1/2} \ll \sqrt{d}$, the bound is better than $O(\sqrt{d}\log(T))$.

The proof of theorems are given in the appendix. It should be noted that although the proof requires a decreasing schedule of α_t and β_{1t} to ensure convergence, numerical experiments show that piecewise constant α_t and constant β_{1t} provide fast convergence speed in practice.

5 EXPRERIMENTS

In this section, we present experiments to evaluate the performance of NAMSG and the OBSB policy for NAMSG, compared with SGD with momentum (Polyak, 1964), CNAG, and popular adaptive

stochastic optimization methods, such as ADAM (Kingma & Ba, 2015), NADAM (Dozat, 2016), and AMSGRAD [1] (Reddi et al., 2018). We study logistic regression and neural networks for multiclass classification, representing convex and nonconvex settings, respectively. The experiments are carried out with MXNET (Chen et al., 2015).

5.1 EXPERIMENTS ON MNIST

We compare the performance of SGD, ADAM, NADAM, CNAG, AMSGRAD, NAMSG and OBSB, for training logistic regression and neural network on the MNIST dataset (LeCun et al., 1998). The dataset consists of 60k training images and 10k testing images in 10 classes. The image size is 28×28.

Logistic regression: In the experiment, the minibatch size is 256. The hyper-parameters for all the methods except NAMSG and OBSB are chosen by grid search (see supplementary materials), and the best results in training are reported. In NAMSG and OBSB, only the step size α is chosen by grid search. We report the train and test results in Figure 2, which are the average of 5 runs. It is observed that OBSB perform the best with respect to train loss, and NAMSG also converges faster than other methods. The test accuracy is roughly consistent with the train loss in the initial epochs, after which they fluctuates for overfitting. The experiment shows that NAMSG and OBSB achieves fast convergence in the convex setting.

Neural networks: In the experiment, we train a simple convolutional neural network (CNN) for the multiclass classification problem on MNIST. The architecture has two 5×5 convolutional layers, with 20 and 50 outputs. Each convolutional layer is followed by Batch Normalization (BN) (Ioffe & Szegedy, 2015) and a 2×2 max pooling. The network ends with a 500-way fully-connected layer with BN and ReLU (Nair & Hinton, 2010), a 10-way fully-connected layer, and softmax. The hyper-parameters are set in a way similar to the previous experiment. The results are also reported in Figure 2, which are the average of 5 runs. We can see that NAMSG has the lowest train loss, which translates to good generalization performance. OBSB also converges faster than other methods. The experiment shows that NAMSG and OBSB are efficient in non-convex problems.

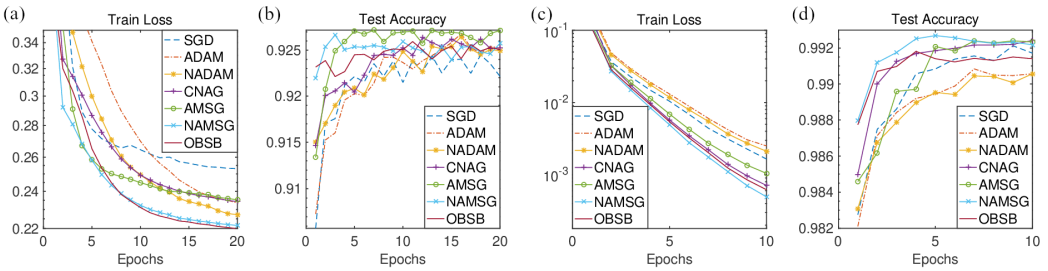

Figure 2: Performance of SGD, ADAM, NADAM, CNAG, AMSG, NAMSG, and OBSB on MNIST. (a) and (b) shows the performance in logistic regression. (c) and (d) compares the results in CNN.

5.2 EXPERIMENTS ON CIFAR-10

In the experiment, we train Resnet-20 (He et al., 2016) on the CIFAR-10 dataset (Krizhevsky, 2009), that consists of 50k training images and 10k testing images in 10 classes. The image size is 32×32.

The architecture of the network is as follows: In training, the network inputs are 28×28 images randomly cropped from the original images or their horizontal flips to save computation. The inputs are subtracted by the global mean and divided by the standard deviation. The first layer is 3×3 convolutions. Then we use a stack of 18 layers with 3×3 convolutions on the feature maps of sizes $\{28, 14, 7\}$ respectively, with 6 layers for each feature map size. The numbers of filters are $\{16, 32, 64\}$ respectively. A shortcut connection is added to each pair of 3×3 filters. The subsampling is performed by convolutions with a stride of 2. Batch normalization is adopted right after each convolution and before the ReLU activation. The network ends with a global average pooling, a 10-way fully-connected layer, and softmax. In testing, the original 32×32 images are used as inputs.

[1] referred to as AMSG in the figures.

We train Resnet-20 on CIFAR-10 using SGD, ADAM, NADAM, CNAG, AMSGRAD, NAMSG, and OBSB. The training for each network runs for 75 epochs. The hyper-parameters are selected in a way similar to the previous experiments, excepting that we divide the constant step size by 10 at the 12000^{th} iteration (in the 62^{th} epoch). A weight decay of 0.001 is used for regularization. Two group of hyper-parameters are obtained for each method, one of which minimizes the train loss before the dropping of step size, and the other maximizes the mean test accuracy of the last 5 epoches.

Figure 3 shows the average results of 5 runs. In experiments to achieve the fastest training speed (Figure 3 (a),(b)), OBSB converges the fastest, and NAMSG is also faster than other methods. Compares with ADAM, OBSB is more than 1 time faster, and NAMSG is roughly 1 time faster to reach the train loss before the dropping of step size. OBSB has the best test accuracy, and NAMSG is better than other methods. CNAG achieves significant acceleration upon SGD, and is also faster than ADAM, NADAM, and AMSGRAD. In experiments to achieve the best generalization (Figure 3 (c),(d)), OBSB still converges the fastest, NAMSG and CNAG converge at almost the same speed, which is faster than other methods. The mean best generalization accuracy of SGD, ADAM, NADAM, CNAG, AMSGRAD, NAMSG, and OBSB are 0.9129, 0.9065, 0.9066, 0.9177, 0.9047, 0.9138, and 0.9132, respectively. CNAG achieves the highest test accuracy. OBSB, NAMSG, and SDG obtains almost the same final test accuracy, which is much higher than ADAM, NADAM, and AMSGRAD. It should be noted that CNAG achieves the best test accuracy at the cost of grid search for 3 parameters, while NAMSG and OBSB only searches for the step size.

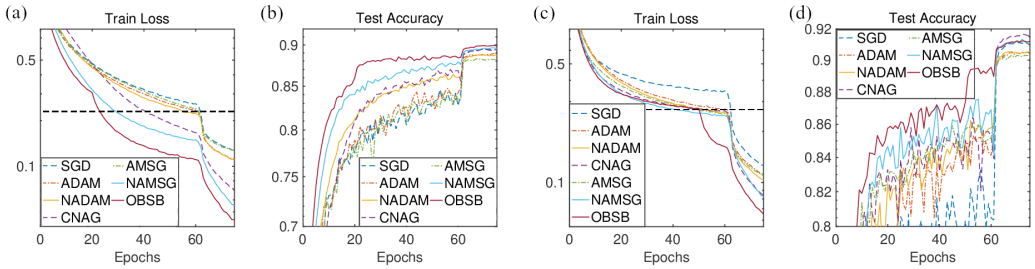

Figure 3: Performance of SGD, ADAM, NADAM, CNAG, AMSG, NAMSG, and OBSB for Resnet-20 on CIFAR-10. (a) and (b) compares the best results for training. (c) and (d) shows the best results for generalization.

The experiments show that in the machine learning problems tested, NAMSG and OBSB converges faster compared with other popular adaptive methods, such as ADAM, NADAM, and AMSGRAD. The acceleration is achieved with low computational overheads and almost no more memory.

6 Conclusions and Discussions

We present the NAMSG method, which computes the gradients at configurable remote observation points, and scales the update vector elementwise by a nonincreasing preconditioner. It is efficient in computation and memory, and is straightforward to implement. A data-dependent regret bound is proposed to guarantee the convergence in the convex setting. The bound is further improved to $O(\log(T))$ for strongly convex functions. The analysis of the optimizing process provides a hyper-parameter policy (OBSB) which leaves only the step size for grid search. Numerical experiments demonstrate that NAMSG and OBSB converge faster than ADAM, NADAM, and AMSGRAD, for the tested problems.

References

D. Amodei, S. Ananthanarayanan, and et al. R. Anubhai. Deep speech 2 : End-to-end speech recognition in English and Mandarin. In *Proceedings of the 33rd International Conference on Machine Learning (ICML 2016)*, pp. 173–182, New York, New York, USA, 2016. Morgan Kaufmann.

L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational Statistics*, pp. 177–186, Heidelberg, 2010. Physica-Verlag HD.

L. Bottou, F. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018. doi: 10.1137/16M1080173.

R. H Byrd, S. L Hansen, J. Nocedal, and Y. Singer. A stochastic quasi-Newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.

Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *CoRR*, abs/1512.01274, 2015. URL http://arxiv.org/abs/1512.01274.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

Timothy Dozat. Incorporating Nesterov momentum into Adam. In *International Conference on Learning Representations*, 2016.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7):257–269, 2011.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, June 2016. doi: 10.1109/CVPR.2016.90.

G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL http://arxiv.org/abs/1502.03167.

Prateek Jain, Sham M. Kakade, Rahul Kidambi, Praneeth Netrapalli, and Aaron Sidford. Accelerating stochastic gradient descent for least squares regression. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet (eds.), *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pp. 545–604. PMLR, 06–09 Jul 2018. URL http://proceedings.mlr.press/v75/jain18a.html.

Rahul Kidambi, Praneeth Netrapalli, Prateek Jain, and Sham M. Kakade. On the insufficiency of existing momentum schemes for stochastic optimization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJTutzbA-.

D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *2015 International Conference on Learning Representations (ICLR 2015)*, pp. 1–11, San Diego, CA, 2015.

A. Krizhevsky. Gradient-based learning applied to document recognition. *Tech Report*, 86(11): 2278–2324, 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

J. Martens. Deep learning via Hessian-free optimization. In *International Conference on Machine Learning (ICML 2010)*, pp. 735–742, 2010.

H. Brendan McMahan and Matthew J. Streeter. Adaptive bound optimization for online convex optimization. *CoRR*, abs/1002.4908, 2010. URL http://arxiv.org/abs/1002.4908.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning*, 2010.

Y. Nesterov. A method of solving a convex programming problem with convergence rate $o(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

J. Nocedal. Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35 (151):773–782, 1980.

B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr Computational Mathematics and Mathematical Physics*, 4(5):791–803, 1964.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of Adam and beyond. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ryQu7f-RZ.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407, 1951.

I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pp. 1139–1147, 2013.

T. Tieleman and G. Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 2012.

Guanghui Wang, Shiyin Lu, Weiwei Tu, and Lijun Zhang. Sadam: A variant of adam for strongly convex functions. 2019.

Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012. URL http://arxiv.org/abs/1212.5701.

A Appendix

A.1 Proof of Theorem 1

In this proof, we use y_i to denote the i^{th} coordinate of a vector y.

From Algorithm 1,

$$\begin{aligned} x_{t+1} =& \Pi_{\mathcal{F}, \sqrt{\hat{V}_t}} \left(x_t - \alpha_t \hat{V}_t^{-1/2} \left(\left(1-\mu_t\right) m_t + \mu_t g_t \right) \right) \\ =& \arg\min_{x \in \mathcal{F}} \left\| \hat{V}_t^{1/4} \left(x - \left(x_t - \alpha_t \hat{V}_t^{-1/2} \left(\left(1-\mu_t\right) m_t + \mu_t g_t \right) \right) \right) \right\| \end{aligned} \tag{A1}$$

Furthermore, $\Pi_{\mathcal{F}, \sqrt{\hat{V}_t}}(x^*) = x^*$ for all $x^* \in \mathcal{F}$. Using Lemma A1 with $\hat{u}_1 = x_{t+1}$ and $\hat{u}_2 = x^*$, we have

$$\begin{aligned} & \left\| \hat{V}_t^{1/4} \left(x_{t+1} - x^* \right) \right\|^2 \leq \left\| \hat{V}_t^{1/4} \left(x_t - \alpha_t \hat{V}_t^{-1/2} \left(\left(1-\mu_t\right) m_t + \mu_t g_t \right) - x^* \right) \right\|^2 \\ =& \left\| \hat{V}_t^{1/4} \left(x_t - x^* \right) \right\|^2 + \alpha_t^2 \left\| \hat{V}_t^{-1/4} \left(\left(1-\mu_t\right) m_t + \mu_t g_t \right) \right\|^2 - 2\alpha_t \left\langle \left(1-\mu_t\right) m_t + \mu_t g_t, x_t - x^* \right\rangle \\ =& \left\| \hat{V}_t^{1/4} \left(x_t - x^* \right) \right\|^2 + \alpha_t^2 \left\| \hat{V}_t^{-1/4} \left(\left(1-\mu_t\right) m_t + \mu_t g_t \right) \right\|^2 \\ & - 2\alpha_t \left\langle \left(1-\mu_t\right) \beta_{1t} m_{t-1} + \left(\mu_t + \left(1-\mu_t\right) \left(1-\beta_{1t}\right) \right) g_t, x_t - x^* \right\rangle \\ \leq & \left\| \hat{V}_t^{1/4} \left(x_t - x^* \right) \right\|^2 + 2\alpha_t^2 \left(\left(1-\mu_t\right)^2 \left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \mu_t^2 \left\| \hat{V}_t^{-1/4} g_t \right\|^2 \right) \\ & - 2\alpha_t \left\langle \left(1-\mu_t\right) \beta_{1t} m_{t-1} + \left(1 - \beta_{1t} + \beta_{1t}\mu_t\right) g_t, x_t - x^* \right\rangle, \end{aligned} \tag{A2}$$

where the second inequality follows from Cauchy-Schwarz inequality.

Since $0 \leq \beta_{1t} < 1$, from the assumptions,

$$0 < 1 - \beta_1 \leq \mu_t = \mu < 1. \tag{A3}$$

Rearrange the inequity (A2), we obtain

$$\begin{aligned}
&\langle g_t, x_t - x^* \rangle \\
\leq &\frac{1}{2\alpha_t \left(1-\beta_{1t}\left(1-\mu\right)\right)} \left(\left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2 - \left\| \hat{V}_t^{1/4} \left(x_{t+1} - x^*\right) \right\|^2 \right) \\
&+ \frac{\alpha_t}{1-\beta_{1t}\left(1-\mu\right)} \left(\left(1-\mu_t\right)^2 \left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \mu_t^2 \left\| \hat{V}_t^{-1/4} g_t \right\|^2 \right) \\
&- \frac{\left(1-\mu_t\right)\beta_{1t}}{1-\beta_{1t}\left(1-\mu\right)} \left\langle m_{t-1}, x_t - x^* \right\rangle \qquad (A4) \\
\leq &\frac{1}{2\alpha_t \left(1-\beta_{1t}\left(1-\mu\right)\right)} \left(\left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2 - \left\| \hat{V}_t^{1/4} \left(x_{t+1} - x^*\right) \right\|^2 \right) \\
&+ \frac{\alpha_t}{1-\beta_{1t}\left(1-\mu\right)} \left(\left(1-\mu_t\right)^2 \left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \mu_t^2 \left\| \hat{V}_t^{-1/4} g_t \right\|^2 \right) \\
&+ \frac{\left(1-\mu_t\right)\beta_{1t}\alpha_t}{2\left(1-\beta_{1t}\left(1-\mu\right)\right)} \left\| \hat{V}_t^{-1/4} m_{t-1} \right\|^2 + \frac{\left(1-\mu_t\right)\beta_{1t}}{2\left(1-\beta_{1t}\left(1-\mu\right)\right)\alpha_t} \left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2,
\end{aligned}$$

where the second inequality also follows from Cauchy-Schwarz inequality, the last equity is due to (A3).

For simplicity, denote

$$\begin{aligned}
P_1 = &\sum_{t=1}^{T} \left(\left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2 - \left\| \hat{V}_t^{1/4} \left(x_{t+1} - x^*\right) \right\|^2 + \left(1-\mu\right)\beta_{1t} \left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2 \right) \\
&/ \left(2\alpha_t \left(1-\beta_{1t}(1-\mu)\right)\right) \\
P_2 = &\sum_{t=1}^{T} \alpha_t \left(\left(1-\mu\right)^2 \left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \mu^2 \left\| \hat{V}_t^{-1/4} g_t \right\|^2 + \frac{1}{2}(1-\mu)\beta_{1t} \left\| \hat{V}_t^{-1/4} m_{t-1} \right\|^2 \right) \\
&/(1-\beta_{1t}(1-\mu)) \qquad (A5)
\end{aligned}$$

Because of the convexity of the objective function, the regret satisfies

$$R_T = \sum_{i=1}^{T} \left(f_t\left(x_t\right) - f_t\left(x^*\right) \right) \leq \sum_{t=1}^{T} \left\langle g_t, x_t - x^* \right\rangle \leq P_1 + P_2 \qquad (A6)$$

The first inequity follows from the convexity of function f_t. The second inequality is due to (A4).

We now bound the term $\sum_{t=1}^{T} \alpha_t \left\| \hat{V}_t^{-1/4} g_t \right\|^2$. We have

$$\begin{aligned}
&\sum_{t=1}^{T} \alpha_t \left\| \hat{V}_t^{-1/4} g_t \right\|^2 = \sum_{t=1}^{T-1} \alpha_t \left\| \hat{V}_t^{-1/4} g_t \right\|^2 + \alpha_T \sum_{i=1}^{d} \frac{g_{T,i}^2}{\sqrt{\hat{v}_{T,i}}} \\
\leq &\sum_{t=1}^{T-1} \alpha_t \left\| \hat{V}_t^{-1/4} g_t \right\|^2 + \alpha_T \sum_{i=1}^{d} \frac{g_{T,i}^2}{\sqrt{v_{T,i}}} \leq \sum_{t=1}^{T-1} \alpha_t \left\| \hat{V}_t^{-1/4} g_t \right\|^2 + \frac{\alpha}{\sqrt{T}} \sum_{i=1}^{d} \frac{g_{T,i}^2}{\sqrt{\left(1-\beta_2\right)\sum_{j=1}^{T} \beta_2^{T-j} g_{j,i}^2}} \\
\leq &\sum_{t=1}^{T-1} \alpha_t \left\| \hat{V}_t^{-1/4} g_t \right\|^2 + \frac{\alpha}{\sqrt{T\left(1-\beta_2\right)}} \sum_{i=1}^{d} \left| g_{T,i} \right| \leq \frac{\alpha}{\sqrt{1-\beta_2}} \sum_{t=1}^{T} \left(\frac{1}{\sqrt{t}} \sum_{i=1}^{d} \left| g_{t,i} \right| \right) \\
\leq &\frac{\alpha}{\sqrt{1-\beta_2}} \sum_{i=1}^{d} \left\| g_{1:T,i} \right\|_2 \sqrt{\sum_{t=1}^{T} \frac{1}{t}} \leq \frac{\alpha\sqrt{1+\log(T)}}{\sqrt{1-\beta_2}} \sum_{i=1}^{d} \left\| g_{1:T,i} \right\|_2 .
\end{aligned} \qquad (A7)$$

In (A7), the third inequity is follows from the definition of v_t , the fifth inequality is due to Cauchy-Schwarz inequality. The final inequality is due to the following bound on harmonic sum: $\sum_{t=1}^{T} 1/t \leq 1 + \log(T)$.

From (A7), and Lemma A2, which bound $\sum_{t=1}^{T} \alpha_t \left\| \hat{V}_t^{-1/4} m_t \right\|^2$, we further bound the term P_2 as

$$
\begin{aligned}
P_2 &\leq \frac{1}{1-\beta_1(1-\mu)} \left(\sum_{t=1}^{T} \alpha_t \left((1-\mu)^2 \left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \frac{1}{2}(1-\mu)\beta_{1t} \left\| \hat{V}_t^{-1/4} m_{t-1} \right\| \right) \right. \\
&\left. + \frac{\alpha\sqrt{1+\log(T)}}{\sqrt{1-\beta_2}} \mu^2 \sum_{i=1}^{d} \left\| g_{1:T_i} \right\|_2 \right) \\
&\leq \frac{1}{1-\beta_1(1-\mu)} \left(\beta_1^2 \sum_{t=1}^{T} \alpha_t \left(\left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \frac{1}{2} \left\| \hat{V}_t^{-1/4} m_{t-1} \right\| \right) \right. \\
&\left. + \frac{\alpha\sqrt{1+\log(T)}}{\sqrt{1-\beta_2}} \mu^2 \sum_{i=1}^{d} \left\| g_{1:T,i} \right\|_2 \right) \\
&\leq \frac{1}{1-\beta_1(1-\mu)} \left(\beta_1^2 \left(\sum_{t=1}^{T} \alpha_t \left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \frac{1}{2} \sum_{t=1}^{T-1} \alpha_t \left\| \hat{V}_t^{-1/4} m_t \right\|^2 \right) \right. \\
&\left. + \frac{\alpha\sqrt{1+\log(T)}}{\sqrt{1-\beta_2}} \mu^2 \sum_{i=1}^{d} \left\| g_{1:T,i} \right\|_2 \right) \\
&\leq \frac{1}{1-\beta_1(1-\mu)} \left(\frac{3}{2}\beta_1^2 \sum_{t=1}^{T} \alpha_t \left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \frac{\alpha\sqrt{1+\log(T)}}{\sqrt{1-\beta_2}} \mu^2 \sum_{i=1}^{d} \left\| g_{1:T,t} \right\|_2 \right) \\
&\leq \left(\frac{3\beta_1^2}{2\left(1-\beta_1\right)\left(1-\gamma\right)} + \mu^2 \right) \frac{\alpha\sqrt{1+\log(T)}}{\left(1-\beta_1(1-\mu)\right)\sqrt{1-\beta_2}} \sum_{t=1}^{d} \left\| g_{1:T,1} \right\|_2 .
\end{aligned}
\tag{A8}
$$

The third inequity is due to $\beta_{1t} \geq \beta_{1t+1}$ and $\hat{v}_{t,i}^{1/2}/\alpha_t \geq \hat{v}_{t-1,i}^{1/2}/\alpha_{t-1}$ by definition.

We also have

$$
\begin{aligned}
&P_1 \\
&\leq \frac{1}{2\alpha_1\left(1-\beta_1(1-\mu)\right)} \left\| \hat{V}_1^{1/4}\left(x_1 - x^*\right) \right\|^2 + \sum_{t=2}^{T} \left(\frac{1}{2\alpha_t\left(1-\beta_{1t}(1-\mu)\right)} \left\| \hat{V}_1^{1/4}\left(x_t - x^*\right) \right\|^2 - \right. \\
&\left. \frac{1}{2\alpha_{t-1}\left(1-\beta_{1t-1}(1-\mu)\right)} \left\| \hat{V}_{t-1}^{1/4}\left(x_t - x^*\right) \right\|^2 \right) + \sum_{t=1}^{T} \frac{\beta_{1t}(1-\mu)}{2\alpha_t\left(1-\beta_{1t}(1-\mu)\right)} \left\| \hat{V}_1^{1/4}\left(x_t - x^*\right) \right\|^2 \\
&\leq \frac{1}{2\alpha_1\left(1-\beta_1(1-\mu)\right)} \left\| \hat{V}_1^{1/4}\left(x_1 - x^*\right) \right\|^2 + \sum_{t=2}^{T} \left(\frac{1}{2\left(1-\beta_{1t}(1-\mu)\right)} \left(\frac{1}{\alpha_t} \left\| \hat{V}_t^{1/4}\left(x_t - x^*\right) \right\|^2 \right. \right. \\
&\left. \left. - \frac{1}{\alpha_{t-1}} \left\| \hat{V}_{t-1}^{1/4}\left(x_t - x^*\right) \right\|^2 \right) + \sum_{t=1}^{T} \frac{\beta_{1t}(1-\mu)}{2\alpha_t\left(1-\beta_{1t}(1-\mu)\right)} \left\| \hat{V}_t^{1/4}\left(x_t - x^*\right) \right\|^2 \right. \\
&\leq \frac{1}{2\left(1-\beta_1(1-\mu)\right)} \left(\frac{1}{\alpha_1} \left\| \hat{V}_1^{1/4}\left(x_1 - x^*\right) \right\|^2 + \sum_{t=2}^{T} \left(\frac{1}{\alpha_t} \left\| \hat{V}_t^{1/4}\left(x_t - x^*\right) \right\|^2 \right. \right. \\
&\left. \left. - \frac{1}{\alpha_{t-1}} \left\| \hat{V}_{t-1}^{1/4}\left(x_t - x^*\right) \right\|^2 \right) + \sum_{t=1}^{T} \frac{\beta_{1t}(1-\mu)}{\alpha_t} \left\| \hat{V}_t^{1/4}\left(x_t - x^*\right) \right\|^2 \right)
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{2\left(1-\beta_1(1-\mu)\right)}\left(\frac{1}{\alpha_1}\sum_{i=1}^{d}\hat{v}_{1,i}^{1/2}\left(x_{1,i}-x_i^*\right)^2+\sum_{t=2}^{T}\left(\sum_{i=1}^{d}\left(x_{t,i}-x_i^*\right)^2\right.\right.\\
&\left.\left.\left(\frac{\hat{v}_{t,i}^{1/2}}{\alpha_t}-\frac{\hat{v}_{t-1,i}^{1/2}}{\alpha_{t-1}}\right)\right)+\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_{1t}(1-\mu)\left(x_{t,i}-x_i^*\right)^2\hat{v}_{t,i}^{1/2}}{\alpha_t}\right)\\
&\leq \frac{D_\infty^2}{2\left(1-\beta_1(1-\mu)\right)}\left(\frac{1}{\alpha_1}\sum_{i=1}^{d}\hat{v}_{1,i}^{1/2}+\sum_{t=2}^{T}\left(\sum_{i=1}^{d}\left(\frac{\hat{v}_{ti}^{1/2}}{\alpha_t}-\frac{\hat{v}_{t-1,i}^{1/2}}{\alpha_{t-1}}\right)\right)+\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_{1t}(1-\mu)\hat{v}_{t,i}^{1/2}}{\alpha_t}\right)\\
&= \frac{D_\infty^2\sqrt{T}}{2\left(1-\beta_1(1-\mu)\right)\alpha}\sum_{i=1}^{d}\hat{v}_{\mathrm{T},i}^{1/2}+\frac{(1-\mu)D_\infty^2}{2\left(1-\beta_1(1-\mu)\right)}\sum_{t=1}^{T}\sum_{i=1}^{d}\frac{\beta_{1t}\hat{v}_{\mathrm{t},i}^{1/2}}{\alpha_t}
\end{aligned}
\tag{A9}
$$

In (A9), the second inequity follows from the assumption $\beta_{1t} < \beta_{1t-1}$, the third and the last inequality is due to $\hat{v}_{t,i}^{1/2}/\alpha_t \geq \hat{v}_{t-1,i}^{1/2}/\alpha_{t-1}$ by definition and the assumption $\alpha_t = \alpha/\sqrt{t}$.

Combining (A6), (A8), and (A9), we obtain

$$
\begin{aligned}
R_T \leq &\frac{1}{\left(1-\beta_1(1-\mu)\right)}\left(\frac{D_\infty^2\sqrt{T}}{2\alpha}\sum_{i=1}^{d}\hat{v}_{\mathrm{T},i}^{1/2}+\frac{(1-\mu)D_\infty^2}{2}\sum_{i=1}^{n}\sum_{i=1}^{d}\frac{\beta_{1t}\hat{v}_{\mathrm{t},i}^{1/2}}{\alpha_i}\right.\\
&\left.+\left(\frac{3\beta_1^2}{2\left(1-\beta_1\right)\left(1-\gamma\right)}+\mu^2\right)\frac{\alpha\sqrt{1+\log(T)}}{\sqrt{1-\beta_2}}\sum_{i=1}^{d}\left\|g_{1T,i}\right\|_2\right).
\end{aligned}
\tag{A10}
$$

The proof is complete.

The Lemmas used in the proof are as follows:

Lemma A1. (McMahan & Streeter, 2010)
For any $Q \in \mathcal{S}_+^d$ and convex feasible set $\mathcal{F} \in R^d$, suppose $\hat{u}_1 = \min_{x\in\mathcal{F}}\left\|Q^{1/2}\left(x-z_1\right)\right\|$ and $\hat{u}_2 = \min_{x\in\mathcal{F}}\left\|Q^{1/2}\left(x-z_2\right)\right\|$ then we have $\left\|Q^{1/2}\left(\hat{u}_1-\hat{u}_2\right)\right\| \leq \left\|Q^{1/2}\left(z_1-z_2\right)\right\|$.

Lemma A2. (Reddi et al., 2018)
For the parameter settings and conditions assumed in Theorem 1, which is the same as Theorem 4 in Reddi et al. (2018), we have

$$
\sum_{t=1}^{T}\alpha_t\left\|\hat{V}_t^{-1/4}m_t\right\|^2 \leq \frac{\alpha\sqrt{1+\log T}}{\left(1-\beta_1\right)\left(1-\gamma\right)\sqrt{1-\beta_2}}\sum_{i=1}^{d}\left\|g_{1:T,i}\right\|_2.
$$

The proofs of Lemma A1 and A2 are described in Reddi et al. (2018).

A.2 Proof of Theorem 2

Because the objective function is strong convex, from (A3) and (A4) the regret satisfies

$$
\begin{aligned}
&R_T = \sum_{i=1}^{T}\left(f_t\left(x_t\right)-f_t\left(x^*\right)\right) \leq \sum_{t=1}^{T}\left(\left\langle g_t, x_t-x^*\right\rangle-\frac{\lambda}{2}\left\|x_t-x^*\right\|^2\right)\\
&\leq \sum_{t=1}^{T}\left(\frac{1}{2\alpha_t\left(1-\beta_{1t}(1-\mu)\right)}\left(\left\|\hat{\nu}_t^{1/4}\left(x_t-x^*\right)\right\|^2-\left\|\hat{V}_t^{1/4}\left(x_{t+1}-x^*\right)\right\|^2\right)-\frac{\lambda}{2}\left\|x_t-x^*\right\|^2\right.\\
&+\frac{\alpha_t}{\left(1-\beta_{1t}(1-\mu)\right)}\left(\beta_1^2\left(\left\|\hat{V}_t^{-1/4}m_t\right\|^2+\frac{1}{2}\left\|\hat{V}_t^{-1/4}m_{t-1}\right\|^2\right)+\mu^2\left\|\hat{V}_t^{-1/4}g_t\right\|^2\right)\\
&\left.+\frac{(1-\mu)\beta_{1t}}{2\alpha_t\left(1-\beta_{1t}(1-\mu)\right)}\left\|\hat{V}_t^{1/4}\left(x_t-x^*\right)\right\|^2\right).
\end{aligned}
\tag{A11}
$$

We divide the righthand side of (A11) to three parts, as

$$
\begin{aligned}
Q_1 =& \sum_{t=1}^{T} \left(\frac{1}{2\alpha_t \left(1-\beta_{1t}(1-\mu)\right)} \left(\left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2 - \left\| \hat{V}_t^{1/4} \left(x_{t+1} - x^*\right) \right\|^2 \right) - \frac{\lambda}{2} \left\| x_t - x^* \right\|^2 \right) \\
Q_2 =& \sum_{t=1}^{T} \frac{\alpha_t}{\left(1-\beta_{1t}(1-\mu)\right)} \left(\beta_1^2 \left(\left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \frac{1}{2} \left\| \hat{V}_t^{-1/4} m_{t-1} \right\|^2 \right) + \mu^2 \left\| \hat{V}_t^{-1/4} g_t \right\|^2 \right) \\
Q_3 =& \sum_{t=1}^{T} \frac{(1-\mu)\beta_{1t}}{2\alpha_t \left(1-\beta_{1t}(1-\mu)\right)} \left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2 .
\end{aligned} \tag{A12}
$$

Firstly, we bound the term Q_1.

$$
\begin{aligned}
Q_1 =& \frac{1}{2\alpha_1 \left(1-\beta_1(1-\mu)\right)} \left\| \hat{V}_1^{1/4} \left(x_1 - x^*\right) \right\|^2 - \frac{1}{2\alpha_t \left(1-\beta_{1t}(1-\mu)\right)} \left\| \hat{V}_t^{1/4} \left(x_{t+1} - x^*\right) \right\|^2 \\
&+ \sum_{t=2}^{T} \left(\frac{1}{2\alpha_t \left(1-\beta_{1t}(1-\mu)\right)} \left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2 - \frac{1}{2\alpha_{t-1} \left(1-\beta_{1t-1}(1-\mu)\right)} \left\| \hat{V}_{t-1}^{1/4} \left(x_t - x^*\right) \right\|^2 \right) \\
&- \sum_{t=1}^{T} \frac{\lambda}{2} \left\| x_t - x^* \right\|^2 \\
\leq& \frac{1}{2\alpha_1 \left(1-\beta_1(1-\mu)\right)} \left\| \hat{V}_1^{1/4} \left(x_1 - x^*\right) \right\|^2 - \frac{\lambda}{2} \left\| x_1 - x^* \right\|^2 \\
&+ \sum_{t=2}^{T} \left(\frac{1}{2\left(1-\beta_{1t}(1-\mu)\right)} \left(\frac{1}{\alpha_t} \left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2 - \frac{1}{\alpha_{t-1}} \left\| \hat{V}_{t-1}^{1/4} \left(x_t - x^*\right) \right\|^2 \right) - \frac{\lambda}{2} \left\| x_t - x^* \right\|^2 \right) \\
=& \frac{1}{2\alpha_1 \left(1-\beta_1(1-\mu)\right)} \left\| \hat{V}_1^{1/4} \left(x_1 - x^*\right) \right\|^2 - \frac{\lambda}{2} \left\| x_1 - x^* \right\|^2 \\
&+ \sum_{t=2}^{T} \left(\frac{1}{2\left(1-\beta_{1t}(1-\mu)\right)} \left(\frac{t}{\alpha} \left\| \hat{V}_t^{1/4} \left(x_t - x^*\right) \right\|^2 - \frac{t-1}{\alpha} \left\| \hat{V}_{t-1}^{1/4} \left(x_t - x^*\right) \right\|^2 \right) - \frac{\lambda}{2} \left\| x_t - x^* \right\|^2 \right) \\
\leq& 0
\end{aligned} \tag{A13}
$$

The first inequity follows from β_t is nonincreasing, the second equity follows from $\alpha_t = \alpha/t$. The last inequity is because of the assumption $\alpha \geq \max_{i\in\{1,\cdots,d\}} \left(t\hat{v}_{t,i}^{1/2} - (t-1,i)\hat{v}_{t-1}^{1/2} \right) / \left(\left(1-\beta_1(1-\mu)\right)\lambda \right)$, and $\epsilon \to 0^+$.

Then, we bound the term Q_2.

$$
\begin{aligned}
Q_2 \leq& \frac{\alpha}{1-\beta_1(1-\mu)} \sum_{t=1}^{T} \frac{1}{t} \left(\beta_1^2 \left(\left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \frac{1}{2} \left\| \hat{V}_t^{-1/4} m_{t-1} \right\|^2 \right) + \mu^2 \left\| \hat{V}_t^{-1/4} g_t \right\|^2 \right) \\
\leq& \frac{\alpha}{1-\beta_1(1-\mu)} \left(\sum_{t=1}^{T} \frac{1}{t} \left(\beta_1^2 \left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \mu^2 \left\| \hat{V}_t^{-1/4} g_t \right\|^2 \right) + \sum_{t=1}^{T-1} \frac{1}{t+1} \frac{\beta_1^2}{2} \left\| \hat{V}_t^{-1/4} m_t \right\|^2 \right) \\
\leq& \frac{\alpha}{1-\beta_1(1-\mu)} \sum_{t=1}^{T} \frac{1}{t} \left(\frac{3}{2} \beta_1^2 \left\| \hat{V}_t^{-1/4} m_t \right\|^2 + \mu^2 \left\| \hat{V}_t^{2-1/4} g_t \right\|^2 \right)
\end{aligned} \tag{A14}
$$

The first inequity is because of $m_0 = 0$, and $\hat{v}_t$ is nondecreasing.

We further bound the two terms in the righthand side of (A14).

$$
\begin{aligned}
&\left\|\hat{V}_t^{-1/4} m_t\right\|^2 \\
\leq &\sum_{i=1}^{d}\left(\sum_{j=1}^{t}\prod_{k=1}^{t-j}\beta_{1(t-k+1)}\right)\left(\sum_{j=1}^{t}\prod_{k=1}^{t-j}\beta_{1(t-k+1)} g_{j,i}^2\right)/\sqrt{(1-\beta_2)\sum_{j=1}^{t}\left(\beta_2^{t-j} g_{t,j}^2\right)} \\
\leq &\sum_{i=1}^{d}\left(\sum_{j=1}^{t}\beta_{1(t-k+1)}^{t-j}\right)\left(\sum_{j=1}^{t}\prod_{k=1}^{t-j}\beta_{1(t-k+1)} g_{j,i}^2\right)/\sqrt{(1-\beta_2)\sum_{j=1}^{t}\left(\beta_2^{t-j} g_{j,i}^2\right)} \\
\leq &\frac{1}{(1-\beta_1)\sqrt{1-\beta_2}}\sum_{i=1}^{d}\left(\sum_{j=1}^{t}\prod_{k=1}^{t-j}\beta_{1(t-k+1)} g_{j,i}^2\right)/\sqrt{\sum_{j=1}^{t}\left(\beta_2^{t-j} g_{j,i}^2\right)} \qquad (A15) \\
\leq &\frac{1}{(1-\beta_1)\sqrt{1-\beta_2}}\sum_{i=1}^{d}\sum_{j=1}^{t}\beta_1^{t-j} g_{j,i}^2/\sqrt{\beta_2^{t-j} g_{j,i}^2} \\
= &\frac{1}{(1-\beta_1)\sqrt{1-\beta_2}}\sum_{i=1}^{d}\sum_{j=1}^{t}\gamma^{t-j}\left|g_{j,i}\right| \\
\leq &\frac{1}{(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)} G_1
\end{aligned}
$$

The first inequality follows from Cauchy-Schwarz inequality.

$$
\begin{aligned}
\left\|\hat{V}_t^{-1/4} g_t\right\|^2 &\leq \sum_{t=1}^{d}\frac{g_{ti}^2}{\sqrt{(1-\beta_2)\sum_{j=1}^{t}\left(\beta_2^{t-j} g_{j,i}^2\right)}} \\
&\leq \sum_{i=1}^{d}\frac{g_{t,i}^2}{\sqrt{1-\beta_2}\left|g_{t,i}\right|} \leq \frac{1}{\sqrt{1-\beta_2}} G_1.
\end{aligned} \qquad (A16)
$$

Combining (A14), (A15), and (A16), we obtain

$$
\begin{aligned}
Q_2 &\leq \frac{\alpha G_1}{(1-\beta_1(1-\mu))\sqrt{1-\beta_2}}\sum_{t=1}^{T}\frac{1}{t}\left(\frac{3}{2}\frac{\beta_1^2}{(1-\beta_1)(1-\gamma)}+\mu^2\right) \\
&\leq \frac{\alpha G_1}{(1-\beta_1(1-\mu))\sqrt{1-\beta_2}}\left(\frac{3}{2}\frac{\beta_1^2}{(1-\beta_1)(1-\gamma)}+\mu^2\right)(\log(T)+1).
\end{aligned} \qquad (A17)
$$

The first inequity follows from the assumptions $\alpha_t = \alpha/t$.

Finally, we bound the term Q_3.

$$
\begin{aligned}
Q_3 &\leq \frac{1}{2(1-\beta_1(1-\mu))}\sum_{t=1}^{T}\left(\frac{(1-\mu)\beta_{1t}}{\alpha_t}\left\|\hat{V}_t^{1/4}(x_t - x^*)\right\|^2\right) \\
&= \frac{(1-\mu)\beta_1}{2\alpha(1-\beta_1(1-\mu))}\sum_{t=1}^{T}\left(\frac{1}{t}\left\|\hat{V}_t^{1/4}(x_t - x^*)\right\|^2\right) \\
&\leq \frac{(1-\mu)\beta_1}{2\alpha(1-\beta_1(1-\mu))}\sum_{t=1}^{T}\frac{1}{t}\sum_{i=1}^{d}\hat{v}_{\mathrm{T},i}^{1/2}(x_{t,i} - x_i^*)^2 \qquad (A18) \\
&\leq \frac{(1-\mu)\beta_1 D_\infty^2}{2\alpha(1-\beta_1(1-\mu))}\sum_{t=1}^{T}\frac{1}{t}\sum_{i=1}^{d}\hat{v}_{t,i}^{1/2} \\
&\leq \frac{(1-\mu)\beta_1 D_\infty^2}{2\alpha(1-\beta_1(1-\mu))}(1+\log(T))\sum_{i=1}^{d}\hat{v}_{T,i}^{1/2}.
\end{aligned}
$$

Algorithm A1 ASGD Algorithm

Input: initial parameter vector x_1, short step $\ddot{\alpha}$, long step parameter $\ddot{\kappa} \geq 1$, statistical advantage parameter $\ddot{\xi} \leq \sqrt{\ddot{\kappa}}$, iteration number T
Output: parameter vector x_T

1: Set $\bar{x}_1 = x_1$, $\ddot{\beta} = 1 - 0.7^2\ddot{\xi}/\ddot{\kappa}$.
2: **for** $t = 1$ to $T - 1$ **do**
3: $\quad g_t = \nabla f_t(x_t)$.
4: $\quad \bar{x}_{t+1} = \ddot{\beta}\bar{x}_t + (1 - \ddot{\beta})\left(x_t - \frac{\ddot{\kappa}\ddot{\alpha}}{0.7} g_t\right)$.
5: $\quad x_{t+1} = \frac{0.7}{0.7+(1-\ddot{\beta})}\left(x_t - \ddot{\alpha} g_t\right) + \frac{1-\ddot{\beta}}{0.7+(1-\ddot{\beta})}\bar{x}_{t+1}$.
6: **end for**

Both the first equity and the first inequity follow from the assumptions $\alpha_t = \alpha/t$ and $\beta_{1t} = \beta_1/t^2$. The last inequity is due to $\hat{v}_{t,i}$ is nondecreasing by definition.

Combining (A11), (A13), (A17), and (A18), we obtain

$$R_T \leq \left(\frac{\alpha G_1}{\sqrt{1-\beta_2}} \left(\frac{3}{2} \frac{\beta_1^2}{(1-\beta_1)(1-\gamma)} + \mu^2 \right) + \frac{(1-\mu)\beta_1 D_\infty^2}{2\alpha} \sum_{i=1}^{d} \hat{v}_{T,i}^{1/2} \right) \frac{1 + \log(T)}{1 - \beta_1(1-\mu)}. \tag{A19}$$

A.3 Equivalence of CNAG and ASGD

The pseudo code of ASGD (Kidambi et al., 2018; Jain et al., 2018) is shown in Algorithm A1. ASGD maintains two iterates: descent iterate x_t and a running average $\bar{x}_t$. The running average is a weighted average of the previous average and a long gradient step from the descent iterate, while the descent iterate is updated as a convex combination of short gradient step from the descent iterate and the running average.

We rewrite the update of Algorithm A1 as

$$\begin{aligned} \begin{bmatrix} \bar{x}_{t+1} \\ x_{t+1} \end{bmatrix} &= \ddot{A} \begin{bmatrix} \bar{x}_t \\ x_t \end{bmatrix} + \ddot{b} g_t \\ \ddot{A} = \begin{bmatrix} \ddot{\beta} & 1 - \ddot{\beta} \\ \frac{(1-\ddot{\beta})\ddot{\beta}}{(1-\ddot{\beta})+0.7} & \frac{(1-\ddot{\beta})^2+0.7}{(1-\ddot{\beta})+0.7} \end{bmatrix}, \ddot{b} &= \begin{bmatrix} \frac{\ddot{\beta}-1}{0.7}\ddot{\kappa}\ddot{\alpha} \\ -\frac{0.7^2+(1-\ddot{\beta})^2\ddot{\kappa}}{0.7((1-\ddot{\beta})+0.7)}\ddot{\alpha} \end{bmatrix}. \end{aligned} \tag{A20}$$

Define the variable transform as

$$\begin{bmatrix} \tilde{m}_t \\ x_t \end{bmatrix} = \ddot{T} \begin{bmatrix} \bar{x}_t \\ x_t \end{bmatrix}, \ddot{T} = \begin{bmatrix} \ddot{l} & \ddot{k}\ddot{l} \\ 0 & 1 \end{bmatrix}, \tag{A21}$$

where $\ddot{k}$ are $\ddot{l}$ are adjustable coefficients.

Combining (A20) and (A21), we obtain

$$\begin{bmatrix} \tilde{m}_{t+1} \\ x_{t+1} \end{bmatrix} = \tilde{T} \begin{bmatrix} \tilde{m}_t \\ x_t \end{bmatrix} + \ddot{T}\ddot{b} g_t, \tilde{T} = \ddot{T}\ddot{A}\ddot{T}^{-1}. \tag{A22}$$

In order to minimize the number of vector computations, we solve the adjustable coefficients $\ddot{k}$ and $\ddot{l}$ by assigning $\tilde{T}_{1,2} = 0, \tilde{T}_{2,1} = 1$. We choose the solution as

$$\ddot{k} = -1, \ddot{l} = \frac{(1-\ddot{\beta})\ddot{\beta}}{(1-\ddot{\beta}) + 0.7}. \tag{A23}$$

Combining (A22) and (A23), we obtain

$$\begin{bmatrix} \tilde{m}_{t+1} \\ x_{t+1} \end{bmatrix} = \tilde{T} \begin{bmatrix} \tilde{m}_t \\ x_t \end{bmatrix} + \ddot{T}\ddot{b} g_t, \tilde{T} = \begin{bmatrix} \frac{0.7\ddot{\beta}}{(1-\ddot{\beta})+0.7} & 0 \\ 1 & 1 \end{bmatrix}. \tag{A24}$$

The update (A24) is identical to the practical form of CNAG update (6) with constant hyper-parameters. The momentum coefficient of CNAG is

$$\beta_t = \beta = \frac{0.7\ddot{\beta}}{(1-\ddot{\beta})+0.7} = (\ddot{\kappa} - 0.49\ddot{\xi})/(\ddot{\kappa} + 0.7\ddot{\xi}), \tag{A25}$$

where the second equity follows from the definition of $\ddot{\beta}$ in Algorithm A1.

It should be noted that the computational overheads of ASGD besides the gradient computation is 6 scalar vector multiplications and 4 vector additions per iteration, while CNAG reduces the costs to 3 scalar vector multiplications and 3 vector additions.

A.4 More details on experiments

We use constant hyper-parameters in the experiments. For ADAM, NADAM, AMSGRAD, and NAMSG, the hyper-parameters $(\alpha, \beta_1, \beta_2)$ are selected from $\{0.0005, 0.001, 0.002, 0.005, 0.01, 0.02\} \times \{0, 0.9, 0.99, 0.999, 0.9999\} \times \{0.99, 0.999\}$ by grid search. For SGD, the hyper-parameters (α, β) are selected from $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1.0, 2.0, 5.0\} \times \{0, 0.9, 0.99, 0.999, 0.9999\}$ by grid search. For CNAG, the hyper-parameters (α, β, μ) are selected from $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1.0, 2.0, 5.0\} \times \{0, 0.9, 0.99, 0.999, 0.9999\} \times \{0.001, 0.01, 0.05, 0.1, 0.2, 0.3, 0.5, 0.9\}$ by grid search. For NAMSG and OBSB, the hyper-parameters (α) is selected from $\{0.0005, 0.001, 0.002, 0.005, 0.01, 0.02\}$ by grid search. (β_1, β_2, μ) are set according to the default values. In OBSB, the grid search runs for 5 epochs in the experiments on MNIST, and 20 epochs in the experiments on CIFAR10. μ is scaled when the converging rate is halved to achieve fast convergence, and at the 50th epoch to maximize generalization.

The experiments are carried out on a workstation with an Intel Xeon E5-2680 v3 CPU and a NVIDIA K40 GPU. The source code of NAMSG can be downloaded at `https://github.com/rationalspark/NAMSG/blob/master/Namsg.py`, and the hyper-paramters can be downloaded at `https://github.com/rationalspark/NAMSG/blob/master/hyperparamters.txt`. The simulation environment is MXNET, which can be downloaded at `http://mxnet.incubator.apache.org`. The MNIST dataset can be downloaded at `http://yann.lecun.com/exdb/mnist`; the CIFAR-10 dataset can be downloaded at `http://www.cs.toronto.edu/~kriz/cifar.html`.