Layerwise and Dimensionwise Local Adaptive Method
for Federated Learning

Anonymous Authors[1]

Abstract

In the emerging paradigm of federated learning
(FL), large amount of clients, such as mobile de-
vices, are used to train possibly high-dimensional
models on their respective data. Due to the low
bandwidth of mobile devices, decentralized op-
timization methods need to shift the computa-
tion burden from those clients to the computation
server while preserving *privacy* and reasonable
communication cost. In this paper, we focus on
the training of deep, as in multi-layered, neural
networks, under the FL settings. We present FED-
LAMB, a novel federated learning method based
on a *layer-wise* and *dimension-wise* updates of the
local models, alleviating the nonconvexity and the
multi-layered nature of the optimization task at
hand. We provide a thorough finite-time conver-
gence analysis for FED-LAMB characterizing
how fast its gradient decreases, which improves
the communication efficiency compared with the
baseline method. We provide experimental re-
sults on various datasets and models, under both
iid and non-iid settings, to show that the proposed
FED-LAMB achieves faster convergence speed
and better generalization performance, compared
to the state-of-the-art.

1. Introduction

A growing and important task while learning models on
observed data, is the ability to train over a large number of
clients which could either be personal devices or distinct en-
tities. In the paradigm of Federated Learning (FL) [10; 18],
a central server orchestrates the optimization over those
clients under the constraint that the data can neither be
centralized nor shared among the clients. This is more com-
putationally efficient, since more computing resources are

---
[1]Anonymous Institution, Anonymous City, Anonymous Region,
Anonymous Country. Correspondence to: Anonymous Author
<anon.email@domain.com>.

used; also, this is a very practical scenario which allows in-
dividual data holders (e.g., mobile devices) to train a model
jointly without leaking the private data. In this paper, as
most modern machine learning tasks, we consider the large
finite-sum optimization problem written as

$$\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} f_i(\theta), \qquad (1)$$

where n denotes the number of workers, f_i represents the
average loss for worker i and θ the global model parameter
taking value in Θ, a subset of $\mathbb{R}^p$. While this formulation
recalls that of standard distributed optimization, the core
principle of FL is different from the traditional distributed
paradigm. FL allows local models to perform multiple local
updates on the local models before the global aggregation.

In this paper, we mainly address two important aspects
of FL: communication efficiency and model performance,
which, in some sense, are also coherent to each other. While
local updates can effectively reduce the number of commu-
nication rounds between the central server and devices, new
techniques are still necessary to tackle the challenge of com-
munication between devices and server, due to, e.g., wireless
bandwidth. Some quantization [1; 27], compression [16]
and sketching [24] methods allow to decrease the number
of bits communicated at each round. Another direction to
improve the communication cost of FL methods is to design
better algorithms to accelerate local training, such that better
local models are sent to the server at each round. This could
lead to reduced communication (to reach a certain accuracy)
as well as possibly improved overall learning performance.
In this work, we will propose an accelerated local model
training framework which achieves both communication
reduction and improved empirical learning performance.

One of the most popular frameworks for FL is called Fed-
SGD [18]: we adopt multiple local Stochastic Gradient De-
scent (SGD) steps in each device, send those local models to
the server that computes the average over those received lo-
cal model parameters, and broadcasts it back to the devices.
As mentioned above, momentum can be added to the lo-
cal SGD training for faster convergence and better learning
performance [30]. In this work, we focus on an alternative
framework as proposed by Chen et al. [2], where AMSGrad,

a popular adaptive optimization method, is deployed locally instead of SGD. Adaptive methods have shown success in many deep learning tasks for fast convergence and good accuracy. Chen et al. [2] showed that when adapted to the FL setting, the so-called "Local AMS" algorithm has communication cost sublinear in R, that is guaranteed to converge to a stationary point with rate $\mathcal{O}(\sqrt{p/Rn})$, where R is the number of communication rounds, p is the model dimensionality and n corresponds to the number of federated clients. Specifically, in Local AMS, each round the global server not only aggregates the local models, but also averages and broadcasts the local second moment estimations, which is a crucial ingredient in AMSGrad controlling the dimensionwise learning rates. Thus, this step can be regarded as a natural remedy to data heterogeneity (i.e., the data on local workers follows different probability distributions), which is a common scenario in practice that affects the performance of FL algorithms [14; 15; 8].

Summary of Contributions. Based on the recent progress in accelerating adaptive methods for efficient training [29], we propose an improved algorithm of Local AMSGrad [2], integrating both dimension-wise and layer-wise adaptive learning rates in each device's local update. Specifically:

- We develop FED-LAMB, a novel optimization framework for federated learning, following a principled layer-wise adaptation strategy to accelerate training of deep networks. The algorithm enjoys fast convergence and good performance from the adaptivity from both AMSGrad and layer-wise learning rate adjustment.

- We provide theoretical analysis on the non-asymptotic convergence rate of FED-LAMB. Our rate, $\mathcal{O}\left(\sqrt{\frac{p}{n}}\frac{1}{\sqrt{hR}}\right)$, where h is the total number of layers and p denotes the dimension, matches the state of the art methods in federated learning and reaches a sublinear convergence in the total number of communication rounds. It also improves the theoretical communication efficiency compared with the baseline Local AMSGrad approach.

- We conduct numerical experiments under both homogeneous and heterogeneous settings on various benchmark datasets such as FMNIST, CIFAR-10 and Tiny-Imagenet. Our results confirm the fast convergence and communication efficiency of the proposed method. In particular, FED-LAMBreaches similar, or better, test accuracy than the baseline local SGD and local AMS methods, with less number of communication rounds.

Roadmap. After establishing a literature review of both realms of federated and adaptive learning in Section 2, we develop in Section 3, our method, namely FED-LAMB,

based on the computation per layer and per dimension, of the adaptive learning rate in the traditional AMSGrad. Theoretical understanding of our method's behaviour with respect to convergence towards a stationary point is developed in Section 4 in nonconvex optimization. We present numerical experiments showing the effectiveness and advantages of our method in Section 5.

2. Related Work

Below, we summarize some relevant works on federated learning and adaptive optimization.

Adaptive gradient methods. Adaptive methods have proven to be the spearhead in many nonconvex optimization tasks. Gradient based optimization algorithms alleviate the possibly high nonconvexity of the objective function by adaptively updating each coordinate of their learning rate using past gradients. Common used examples include RMSprop [26], Adadelta [31], Adam [9], Nadam [4] and AMSGrad [22]. Their popularity and efficiency are due to their great performance at training deep neural networks. They generally combine the idea of adaptivity from AdaGrad [5; 17], as explained above, and the idea of momentum from Nesterov's Method [19] or Heavy ball method [20] using past gradients. AdaGrad displays a great edge when the gradient is sparse compared to other classical methods. The anisotropic nature of this update represented a real breakthrough in the training of high dimensional and nonconvex loss functions. This adaptive learning rate helps accelerate the convergence when the gradient vector is sparse [5]. Yet, when applying AdaGrad to train deep neural networks, it is observed that the learning rate might decay too fast, see Kingma & Ba [9] for more details. Consequently, [9] develops Adam leveraging a moving average of the gradients divided by the square root of the second moment of this moving average (element-wise multiplication). A variant, called AMSGrad described in Reddi et al. [22], ought to fix Adam failures using a max operator. Beyond improving the convergence speed of optimization methods, several studies including Zhou et al. [35] also focus on improving their generalization properties. A natural extension of Adam has been developed in You et al. [29] specifically for multi layered neural network where a principled layer-wise adaptation strategy is employed to accelerate training of deep neural networks using large mini-batches using either SGD or adaptive method under the setting of a classical single server. In simple terms, the idea is based on the observation that in a large deep neural network, the magnitude of the gradient might be too small in comparison with the magnitude of the weight for some layers of the model, hence slowing down the overall convergence. As a consequence, layer-wise adaptive learning rate is applied, such that in each iteration the model can move sufficiently far. This

method empirically speeds up the convergence significantly in classical sequential models and can be provably faster than baseline methods.

Federated learning. An extension of the well known parameter server framework, where a model is being trained on several servers in a distributed manner, is called federated learning (FL), see Konečnỳ et al. [10]. Here, the central server only plays the role of computing power for aggregation and global update of the model. Compared with the distributed learning paradigm, in federated learning, the data stored in each worker must not be seen by the central server – preserving privacy is key – and the nature of those workers (e.g., mobile devices), combined with their usually large amount, makes communication between the devices and the central server less appealing – communication cost needs to be controlled. Thus, while traditional distributed gradient methods [21; 13; 32] do not respect those constraints, it has been proposed in McMahan et al. [18], an algorithm called Federated Averaging – Fed-SGD – extending parallel SGD with local updates performed on each device. In Fed-SGD, each worker updates their own model parameters locally using SGD, and the local models are synchronized by periodic averaging on the central parameter server.

3. Layer-wise and Dimension-wise Adaptive Optimization

Notations. We denote by θ the vector of parameters taking values in $\mathbb{R}^d$. For each layer $\ell \in [\![\mathsf{h}]\!]$, where h is the total number of layers of the neural networks, and each coordinate $j \in [\![p_\ell]\!]$ where p_ℓ is the dimension per layer ℓ ($p := \sum_{\ell=1}^{\mathsf{h}} p_\ell$ denotes the total dimension). We also note $\theta_{r,i}^{\ell,t}$ its value for layer ℓ at round r, local iteration t and for worker i. The gradient of f with respect to θ^ℓ is denoted by $\nabla_\ell f(\theta)$. The smoothness per layer is denoted by L_ℓ for each layer $\ell \in [\![\mathsf{h}]\!]$.

3.1. Local AMSGrad and Periodic Averaging

Under the federated learning setting, we stress on the importance of reducing, at each round, the communication cost between the central server, used mainly for aggregation purposes, and the many clients used for gradient computation and local updates. Using Periodic Averaging after few local epochs, updating local models on each device, as developed in McMahan et al. [18] is the gold standard for achieving such communication cost reduction. Intuitively, one rather shifts the computation burden from the many clients to the central server as much as possible. This technique allows for fewer local epochs and a better global model, from a loss minimization (or model fitting) perspective. The premises of that new paradigm are SGD updates performed locally on each device then averaged periodically, see Konečnỳ et al.

[10]; Zhou & Cong [34]. The heuristic efficiency of local updates using SGD and periodic averaging has been studied in Stich [25]; Yu et al. [30] and shown to reach a similar sublinear convergence rate as in the standard distributed optimization settings. Then, with the growing need of training far more complex models, e.g., deep neural networks, several efficient methods, built upon adaptive gradient algorithms, such as Local AMSGrad in Chen et al. [2], extended both empirically and theoretically the benefits of performing local updates coupled with periodic averaging, especially when dealing with non-iid local data distribution making it a better alternative than local SGD on many learning tasks.

3.2. Layer-wise and Dimension-wise Adaptive Local Update
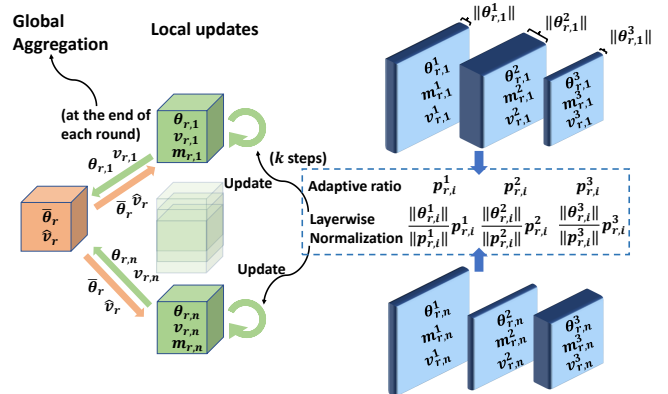

Figure 1: Illustration of FED-LAMB (Algorithm 1), with a three-layer network and $\phi(x) = x$ as an example. For device i and each local iteration in round r, the adaptive ratio of j-th layer $p_{r,i}^j$ is normalized according to $\|\theta_{r,i}^j\|$, and then used for updating the local model. At the end of each round r, local worker i sends $\theta_{r,i} = [\theta_{r,i}^\ell]_{\ell=1}^{\mathsf{h}}$ and $v_{r,i}$ to the central server, which transmits back aggregated θ and $\hat{v}$ to devices to complete a round of training.

For training large deep neural networks, [29] proposed LAMB, an acceleration framework for both SGD and adaptive algorithm that allows large-batch training of BERT in hours. The idea is based on the observation that, during the training of large neural networks, different layers may have very different (absolute) scales, while the magnitude of the gradients in these layers may be similar. Intuitively, this means that in some iterations, the parameter may move a too small step even when the direction (negative gradient) is correct. Therefore, [29] proposed to normalize the gradients in each layer according to the scale of the network layer. Effectively, the algorithm assigns different learning rates to different layers. Empirically, this strategy significantly accelerates the convergence for training large deep models.

Inspired by the merits of local adaptive methods and the success of layer-wise adaptive learning rates, we propose a layer-wise and dimension-wise local AMS algorithm which

Algorithm 1 FED-LAMB

1: **Input**: parameter $0 < \beta_1, \beta_2 < 1$, and learning rate α_t, weight decaying parameter $\lambda \in [0, 1]$.

2: **Initialize**: $\theta_{0,i} \in \Theta \subseteq \mathbb{R}^d$, $m_{0,i}^0 = \hat{v}_{0,i}^0 = v_{0,i}^0 = 0$, $\forall i \in [\![n]\!]$, and $\bar{\theta}_0 = \frac{1}{n} \sum_{i=1}^n \theta_{0,i}$.

3: **for** $r = 1$ to R **do**

4:    **for** parallel for device $i \in D^r$ **do**

5:       Set $\theta_{r,i}^0 = \bar{\theta}_{r-1}$.

6:       Set $m_{r,i}^0 = m_{r-1,i}^T$, $v_{r,i}^0 = \hat{v}_{r-1}$.

7:       **for** $t = 1$ to T **do**

8:          Compute stochastic gradient $g_{r,i}^t$ at $\theta_{r,i}^0$.

9:          $m_{r,i}^t = \beta_1 m_{r,i}^{t-1} + (1 - \beta_1) g_{r,i}^t$

10:         $m_{r,i}^t = m_{r,i}^t / (1 - \beta_1^t)$.

11:         $v_{r,i}^t = \beta_2 v_{r,i}^{t-1} + (1 - \beta_2)(g_{r,i}^t)^2$

12:         $v_{r,i}^t = v_{r,i}^t / (1 - \beta_2^t)$.

13:         Compute the ratio $p_{r,i}^t = m_{r,i}^t / (\sqrt{\hat{v}_r^t} + \epsilon)$.

14:         Update local model for each layer $\ell \in [\![h]\!]$:

$$\theta_{r,i}^{\ell,t} = \theta_{r,i}^{\ell,t-1} - \frac{\alpha_r \phi(\|\theta_{r,i}^{\ell,t-1}\|)(p_{r,i}^{\ell,t} + \lambda \theta_{r,i}^{\ell,t-1})}{\|p_{r,i}^{\ell,t} + \lambda \theta_{r,i}^{\ell,t-1}\|} . \tag{2}$$

15:       **end for**

16:       Devices send $\theta_{r,i}^T = [\theta_{r,i}^{\ell,T}]_{\ell=1}^h$ and $v_{r,i}^T$ to server.

17:    **end for**

18:    Server computes averages of the local models $\bar{\theta}_r = [\bar{\theta}_r^{\ell,T}]_{\ell=1}^h = [\frac{1}{n} \sum_{i=1}^n \theta_{r,i}^{\ell,T}]_{\ell=1}^h$ and $\hat{v}_{r+1} = \max(\hat{v}_r, \frac{1}{n} \sum_{i=1}^n v_{r,i}^T)$ and send them back to the devices.

19: **end for**

20: **Output**: Global model parameter $\bar{\theta}_R = [\bar{\theta}_R^{\ell,T}]_{\ell=1}^h$.

---

is detailed in Algorithm 1 and depicted in Figure 1. The proposed algorithm is a natural adaptation of the vanilla AMSGrad method, for multi-layer neural networks under the federated learning setting. Here, "multi-layer" includes a broad class of network architectures that can be parameterized layer-by-layer (e.g., CNN, ResNet, Transformers). Lines 8-11 in Algorithm 1 correspond to the standard first and second moment estimation and correction performed by AMSGrad locally. The local model update rule (2) in Line 12 incorporates the layer-wise normalization according to the magnitude of each layer's weights. For example, taking ϕ as the identity function and weight decay $\lambda = 0$, then the model θ is updated by $-\frac{\alpha \|\theta\|}{\|p\|} \cdot p$, instead of $-\alpha p$ as in the standard AMSGrad update. The norm of each update is precisely $\alpha \|\theta\|$. That is, we force the change in each layer's parameter to have norm proportional to the scale of the layer's weight. This is a novel extension of the layer-wise adaptivity to the federated learning framework. Therefore, the proposed FED-LAMB exhibits two-fold adaptivity—a dimension-wise adaptive learning rate with respect to the

square root of the second moment used in AMSGrad, and a layer-wise adaptive normalization brought by LAMB. Such two-fold adaptivity has not been considered in FL literature, and we will demonstrate the advantage of this scheme theoretically and empirically in the remainder of the paper.

4. Convergence of FED-LAMB

We now develop the theoretical analysis of Algorithm 1. Based on classical stochastic nonconvex optimization results, we derive a collection of bounds aiming to provide a better understanding of the convergence behavior of our distributed optimization method under the federated learning framework. The main challenges we ought to overcome are: (i) The large amount of decentralized workers working solely on their own data stored locally. (ii) A periodic averaging occurs on the central server pushing each of those clients to send local models after some local iterations. (iii) Each client computes a backpropagation of the main model, i.e., the DNN, and then updates its local version of the model via an adaptive gradient method: the distinctiveness being that those updates are done *dimension-wise* and *layer-wise*. Our analysis encompasses those challenges and leads to an informative convergence rates depending on the quantities of interest in our problem: the number of layers of the neural network, the number of communications rounds and the number of active clients.

4.1. Finite Time Analysis of FED-LAMB

In the sequel, the analysis of our scheme we provide is *global*, in the sense that it does not depend on the initialization of our algorithm, and *finite-time*, meaning that it is true for any arbitrary number of communication rounds, in particular small ones. In the particular context of nonconvex stochastic optimization for distributed clients, we assume the following:

H1. *(Smoothness per layer) For $i \in [\![n]\!]$ and $\ell \in [\![L]\!]$:* $\left\| \nabla f_i(\theta^\ell) - \nabla f_i(\vartheta^\ell) \right\| \le L_\ell \left\| \theta^\ell - \vartheta^\ell \right\|.$

We add some classical assumption on the gradient of the objective function:

H2. *(Unbiased and Bounded gradient) The stochastic gradient is unbiased for any iteration $r > 0$: $\mathbb{E}[g_r] = \nabla f(\theta_r)$ and is bounded from above, i.e., $\|g_t\| \le M$.*

H3. *(Bounded variance) The variance of the stochastic gradient is bounded for any iteration $r > 0$ and any dimension $j \in [\![d]\!]$: $\mathbb{E}[|g_r^j - \nabla f(\theta_r)^j|^2] < \sigma^2$.*

H4. *(Bounded Scale) For any $a \in \mathbb{R}_+^*$, there exists strictly positive constants such that $\phi_m \le \phi(a) \le \phi_M$.*

We now state our main result regarding the non asymptotic convergence analysis of our Algorithm 1 for multiple local updates and true for any communication rounds number

R.

Theorem 1. *Assume **H1-H4**. Consider $\{\overline{\theta_r}\}_{r>0}$, the sequence of parameters obtained running Algorithm 1 with a constant learning rate α. Let the number of local epochs be $T \geq 1$ and $\lambda = 0$. Then, for any round $R > 0$, we have*

$$\frac{1}{R}\sum_{r=1}^{R}\mathbb{E}\left[\left\|\frac{\nabla f(\overline{\theta_r})}{\hat{v}_r^{1/4}}\right\|^2\right]$$

$$\leq \sqrt{\frac{M^2 p}{n}}\frac{\triangle}{\mathsf{h}\alpha R} + \frac{4\alpha\alpha^2 LM^2(T-1)^2\phi_M^2(1-\beta_2)p}{\sqrt{v_0}}$$

$$+ 4\alpha\frac{M^2}{\sqrt{v_0}} + \frac{\phi_M\sigma^2}{Rn}\sqrt{\frac{1-\beta_2}{M^2 p}} + 4\alpha\left[\phi_M\frac{\mathsf{h}\sigma^2}{\sqrt{n}}\right]$$

$$+ 4\alpha\left[\phi_M^2\sqrt{M^2 + p\sigma^2}\right] + cst,$$

$$(3)$$

where $\triangle = \mathbb{E}[f(\bar{\theta}_1)] - \min_{\theta\in\Theta} f(\theta)$.

Two important Lemmas are required in the proof of the Theorem above. We also report the complete proof of our bound in the Appendix of this paper. The first result gives a characterization of the gap between the averaged model, that is computed by the central server in a periodic manner, and each of the local models stored in each client $i \in [\![n]\!]$.

Lemma 1. *Consider $\{\overline{\theta_r}\}_{r>0}$, the sequence of parameters obtained running Algorithm 1. Then for $i \in [\![n]\!]$ and $r > 0$, the gap $\|\overline{\theta_r} - \theta_{r,i}\|^2$ satisfies:*

$$\|\overline{\theta_r} - \theta_{r,i}\|^2 \leq \alpha_r^2 M^2\phi_M^2\frac{(1-\beta_2)p}{v_0},$$

where ϕ_M is defined in H4 and p is the total number of dimensions $p = \sum_{\ell=1}^{\mathsf{h}} p_\ell$.

The gap is provably bounded by some quantities of interest such as the total dimension of the multi-layered model p, the learning rate and the assumed upper bound of the gradient, see H2.

Lemma 2. *Consider $\{\overline{\theta_r}\}_{r>0}$, the sequence of parameters obtained running Algorithm 1. Then for $r > 0$:*

$$\left\|\frac{\overline{\nabla}f(\theta_r)}{\sqrt{v_r^t}}\right\|^2 \geq \frac{1}{2}\left\|\frac{\nabla f(\overline{\theta_r})}{\sqrt{v_r^t}}\right\|^2 - \overline{L}\alpha^2 M^2\phi_M^2\frac{(1-\beta_2)p}{v_0},$$

where M is defined in H2, $p = \sum_{\ell=1}^{\mathsf{h}} p_\ell$ and ϕ_M is defined in H4.

Note that the end goal is to characterize how fast the gradient of the averaged/global parameter $\overline{\theta_r}$ goes to zero, but not

the averaged gradient. Lemma 2 allows us to convert the suboptimality condition $\left\|\frac{\overline{\nabla}f(\theta_r)}{\sqrt{v_r^t}}\right\|$ to the desired one which is $\left\|\frac{\nabla f(\overline{\theta_r})}{\sqrt{v_r^t}}\right\|$. By choosing a suitable decreasing learning rate, we have the following simplified result.

Corollary 1. *Under the same setting as Theorem 1, with $\alpha = \mathcal{O}(\frac{1}{\sqrt{\mathsf{h}R}})$, it holds that*

$$\frac{1}{R}\sum_{r=1}^{R}\mathbb{E}\left[\left\|\frac{\nabla f(\overline{\theta_r})}{\hat{v}_t^{1/4}}\right\|^2\right] \qquad (4)$$

$$\leq \mathcal{O}\left(\sqrt{\frac{p}{n}}\frac{1}{\sqrt{\mathsf{h}R}} + \frac{\sigma^2}{Rn\sqrt{p}} + \frac{(T-1)^2 p}{\mathsf{h}^3 R^{3/2}}\right).$$

The leading two terms display a dependence of the convergence rate of FED-LAMB on the initialization and the variance of the stochastic gradient (see H3), which are common in distributed optimization. The last term involves the number of local updates which relates to the communication efficiency. More discussion will be provided next.

4.2. Comparisons

We dedicate the following paragraph to a discussion on the bound (and implications) derived above in comparison with known results most relevant to our interest in literature.

LAMB bound in You et al. [29]: We first start our discussion with the comparison of convergence rate of FED-LAMB with that of LAMB, Theorem 3 in You et al. [29]. The convergence rates of FED-LAMBand LAMB differ in two ways: (i) First, note that the characterization, on the suboptimality, or convergence criterion, is given at the averaged parameters noted $\overline{\theta}_r$ due to our distributed settings. It is thus natural to consider the evolution of our objective function, precisely its gradient, evaluated at some global model values –as opposed to the outcome of a single step drift in the central server paradigm. Besides, for ease of interpretation, the LHS of (3) is summed over all rounds instead of a fictive random termination point. A simple calculation would lead to such characterization found in several nonconvex stochastic optimization paper such as [6]. (ii) Assuming that the convergence criterion in both Theorems is of similar order (which happens for a large enough number of rounds), the convergence rate of FED-LAMB displays a similar $\mathcal{O}(1/R)$ behaviour for the initialization term. That said, despite the distributed (federated) setting, our dimension-wise and layer-wise method benefits from the double adaptivity phenomenon explained above and exhibited in LAMB [29], under a central server setting.

Fed-AMS bound in Chen et al. [2]: We now discuss the similarities and differences between Fed-AMS, the baseline

distributed adaptive method developed in Chen et al. [2], and our FED-LAMB. For clarity, we restate their main result (Theorem 2) under our notations.

> **Theorem 2** (Corollary 5.2 in Chen et al. [2]). *Under some regularity conditions on the local losses and similar assumption as ours, with some properly chosen learning rate, when $R \geq \mathcal{O}(T^3\sqrt{n})$, Fed-AMS has convergence rate*
>
> $$\frac{1}{R}\sum_{r=1}^{R}\mathbb{E}\left[\left\|\frac{\nabla f(\overline{\theta_r})}{\sqrt{v_r}}\right\|^2\right] \leq \mathcal{O}(\frac{\sqrt{p}}{\sqrt{nR}}). \qquad (5)$$

Firstly, when the number of rounds R is sufficiently large, both rates (4) and (5) are dominated by $\mathcal{O}(\frac{\sqrt{p}}{\sqrt{nR}})$, matching the convergence rate of the standard AMSGrad (e.g. [33]). The acceleration of our layer-wise scheme is exhibited in the $\mathcal{O}(1/(nR))$ term and the dependence on the number of layers $\mathcal{O}(1/(h^3R^{3/2}))$ term. Note that the boundedness assumption is on each dimension in H3 and leads to a manifestation of the term $\sqrt{p}$ in both rates. This dependency on p can be removed when H3 is assumed globally, which is also common in optimization literature.

Secondly, in (4), the last term containing the number of local updates T is small as long as $T \leq \mathcal{O}(\frac{R^{1/2}h^{5/4}}{(np)^{1/4}})$. Treating $p^{1/4}/h = \mathcal{O}(1)$, the result implies that for a given T, we can get the same rate of convergence as vanilla AMSGrad with $R \geq \mathcal{O}(T^2\sqrt{n})$ rounds of communication. From Theorem 2, we know that Fed-AMS requires $R \geq \mathcal{O}(T^3\sqrt{n})$ rounds to achieve the same rate. This implies that, our FED-LAMB reduces the number of communication rounds needed to reach an ϵ-stationary point compared with [2]. Therefore, by leveraging the layer-wise acceleration on local models, our FED-LAMB improve the communication cost of Fed-AMS.

4.3. More Discussions

We provide more discussion on the algorithmic and theoretical properties of FED-LAMB from the following aspects:

Communication and Client Efficiency: The (sublinear) dependence on the number of communication rounds of our bound matches that of most recent methods in federated learning, e.g. SCAFFOLD [8], a solution to the problem posed by heterogeneity of the data in each client. Yet, contrary to SCAFFOLD, our method only sends bits once per communication round while SCAFFOLD needs to send two vectors, including an additional control variate term from the clients to the central server. Our result also matches the communication bound of [23] which adapts Adam [9] to the federated setting. However, the algorithm of [23] performs adaptive updates only at the central server, while SGD is still used for local updates. In addition, the $1/\sqrt{n}$ term in

convergence rate of FED-LAMB implies a linear speedup effect in the number of local workers, which also matches the dependency on n of most federated learning methods.

Data Heterogeneity: To demonstrate the effect of non-iid data distribution theoretically, some related works pose assumptions on the global variance and the local variance of the stochastic gradients of the objective function (1) separately, such that both variances appears in the convergence rate. Our analysis can also be easily extended to incorporate the global variance term. While some works including Karimireddy et al. [8] target on designing specific strategies to alleviate the negative influence of data heterogeneity, we note that our FED-LAMB are, in some sense, naturally capable of balancing the heterogeneity in different local data distributions. As mentioned before, this is largely due to the "moment averaging" step (line 16 in Algorithm 1), where the adaptive learning rates guided by the second moment estimation are aggregated among local workers periodically. In our experiments, we highlight that the advantage of FED-LAMB is consistent under both homogeneous and heterogeneous settings.

5. Numerical Experiments

In this section, we conduct numerical experiments on various datasets and network architectures to justify the effectiveness of our proposed method in practice. Our main objective is to validate the benefit of dimension-wise adaptive learning when integrated with the locally adaptive FL method. We observe that our method empirically confirms its edge in terms of convergence speed. Basically, FED-LAMB reduces the number of rounds and thus the communication cost required to achieve a similar stationary point (or test accuracy) than the baseline methods. In many cases, FED-LAMB could also bring notable improvement in the generalization performance.

Methods and tuning. We evaluate four FL algorithms:

1. Fed-SGD [18], standard federated averaging with local SGD updates.

2. Fed-AMS [2], locally adaptive AMSGrad.

3. Adp-Fed (*Adaptive Federated Optimization*), the federated adaptive algorithm proposed by [23]. Adp-Fed performs local SGD updates. At each round r, the changes in local models, $\triangle_i = w_{r,i}^T - w_{r,i}^0$, $i = 1, ..., n$, are sent to the central server for an aggregated Adam update. See Appendix A for a precise description.

4. Our proposed FED-LAMB (Algorithm 1), layer-wise accelerated local AMSGrad.

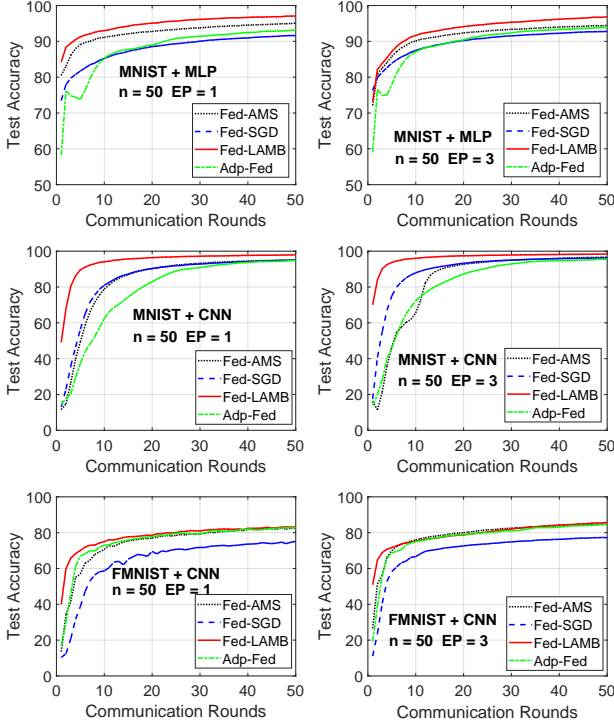For all the adaptive methods, we set hyper-parameters

Figure 2: **i.i.d. data setting**. Test accuracy on MNIST and FMNIST against the number of communication rounds with 50 clients. (Left) 1 local epoch. (Right) 3 local epochs. **1st row:** MNIST+MLP. **2nd row:** MNIST + CNN. **3rd row:** FMNIST + CNN. We see that FED-LAMB converges faster to better solution (higher test accuracy) in all cases.
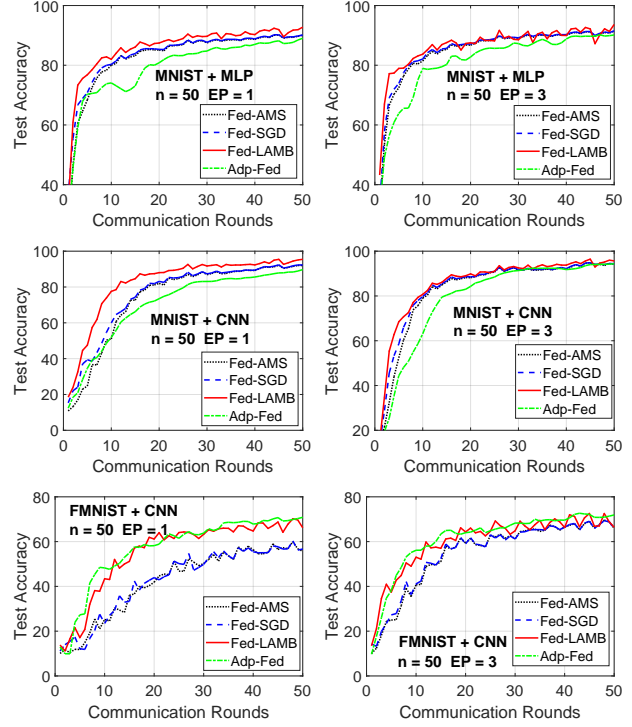
Figure 3: **non-i.i.d. data setting.** Test accuracy on MNIST and FMNIST against the number of communication rounds, with 50 clients. (Left) 1 local epoch. (Right) 3 local epochs. **1st row:** MNIST+MLP. **2nd row:** MNIST + CNN. **3rd row:** FMNIST + CNN. We see that FED-LAMB converges faster to better solution (higher test accuracy) in all cases.

$\beta_1 = 0.9$, $\beta_2 = 0.999$ as the recommended default [22]. Regarding the federated learning environment, we use 50 local workers with 0.5 participation rate. That is, we randomly pick half of the workers to be active for training in each round. We set the local mini-batch size as 128. In each round, the training samples are allocated to the active devices, and one local epoch is finished after all the local devices run one epoch over their received samples by mini-batch training.

We tune the initial learning rate α for each algorithm over a fine grid. For Adp-Fed, there are two learning rates involved. We tune the combination of local learning rate (for local SGD) and global learning rate (for global Adam) over a fine grid. For FED-LAMB the parameter λ in Algorithm 1 controlling the overall scale of the layer-wise gradients is tuned from $\{0, 0.01, 0.1\}$, and use the identity function for $\phi(\cdot)$. The detailed parameter tuning diagram can be found in Appendix A. For each run, we report the test accuracy with the best α and λ. The results are averaged over three independent runs, with same initialization for every method.

Datasets and models. We run our experiments on four popular benchmark image classification datasets: MNIST [12], Fashion MNIST (FMNIST) [28], CIFAR-10 [11] and Tiny-

ImageNet [3]. The MNIST dataset contains 60000 training samples and 10000 test samples, from 10 classes of handwritten digits from 0 to 9. The FMNIST dataset has the same sample size and train/test split as MNIST, but the samples are fashion products (e.g., dress, bags) which makes it harder to train than MNIST. The CIFAR-10 dataset has 50000 training images and 10000 test images, from 10 classes. The TinyImageNet is a subset of the ImageNet dataset. It includes 100000 64×64 images for from 200 classes for training and 10000 for testing. For MNIST, we apply 1) a simple multi-layer perceptron (MLP), which has one hidden layer containing 200 cells with dropout; 2) Convolutional Neural Network (CNN), which has two max-pooled convolutional layers followed by a dropout layer and two fully-connected layers with 320 and 50 cells respectively. This CNN is also implemented for FMNIST. For CIFAR-10 and TinyImageNet, we use ResNet-18 [7].

5.1. Comparison under iid setting

In Figure 2, we report the test accuracy of MLP trained on MNIST, as well as CNN trained on MNIST and FMNIST, where the data is i.i.d. allocated among the clients. We test 1 local epoch and 3 local epochs (more local iterations). In all the figures, we observe a clear advantage of FED-LAMB

over the competing methods in terms of the convergence speed. In particular, we can see that FED-LAMB is able to achieve the same accuracy with fewest number of communication rounds, thus improving the communication efficiency. For instance, this can be observed as follows: on MNIST + CNN (1 local epoch), Fed-AMS requires 20 rounds to achieves 90% accuracy, while FED-LAMB only takes 5 rounds. This implies a 75% reduction in the communication cost. Moreover, on MNIST, FED-LAMB also leads to improved generalisation performance, i.e., test accuracy. Note that, the result on MLP to a good extent provides a straightforward illustration on the benefit of layer-wise adaptivity in FED-LAMB since compared with Fed-AMS, the only difference is that the learning rate becomes adaptive to the scale of the single hidden layer in FED-LAMB. Lastly, we see a faster convergence with 3 local epochs. For iid data distribution, this is expected and consistent with previous results in literature (e.g., [23]).

5.2. Comparison under non-iid setting

In Figure 3, we provide the results on MNIST and FMNIST dataset, when the local data has non-iid distribution (i.e., under data heterogeneity). In particular, in each round of federated training, every local device only receives samples from one class (out of ten). This is known to be the scenario where federated learning is harder to generalize well [18], thus an important case for the empirical evaluation of FL methods. First of all, from Figure 3, we see that for experiments with 1 local epoch, in all cases our proposed FED-LAMB outperforms the three baseline methods. Similar to the plots for iid data setting, FED-LAMB provides faster convergence speed and achieves higher test accuracy than Fed-SGD and Fed-AMS. The advantage is especially significant for the CNN model, e.g., it improves the accuracy of Fed-SGD and Fed-AMS by more than 10% on FMNIST. On both two datasets, FED-LAMB saves around 50% communication rounds to reach a same accuracy level as Fed-AMS. The other baseline method, Adp-Fed, performs as good as our FED-LAMB on FMNIST, but worse than other methods on MNIST. The relative comparison is basically the same when we conduct 3 local epochs. Yet, the advantage of FED-LAMB becomes less significant than what we observed in Figure 2 with iid local data distribution. One plausible reason is that when the local data is highly non-iid as in our case, the fast convergence of the local models as in FED-LAMB might not be as beneficial when we allow too many local updates per round. Intuitively speaking, learning the local models "too fast" might not be a good thing to the globally aggregated model, since each local model is trained only with a few classes of data points, i.e., local models target at different loss functions.

In Figure 4, we present the experiment results on CIFAR-10 and TinyImageNet datasets trained by ResNet-18. When
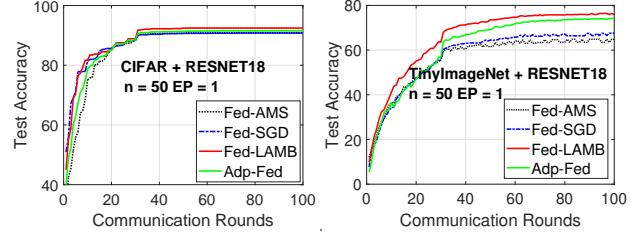

Figure 4: **non-i.i.d. data setting.** Test accuracy on CIFAR-10 + ResNet-18 and TinyImagenet + ResNet-18 with 50 clients.

training these two models, we decrease the learning rate by 10 at 30 and 70 communication rounds. From Figure 4, we can draw similar conclusion as before: the proposed FED-LAMB is the best method in terms of both convergence speed and generalization accuracy. In particular, on TinyImageNet, we see that FED-LAMB has a significant advantage over all three baselines. Although Adp-Fed performs better than Fed-SGD and Fed-AMS, it is considerably worse than FED-LAMB. For reference, we also report the overall test accuracy at the end of training in Table 1. FED-LAMB achieves the highest accuracy on both datasets. On TinyImagenet, FED-LAMB reaches 76% after 100 communication rounds, against around 65% for Fed-AMS, 68% for Fed-SGD and 74% for Adp-Fed.

Table 1: Test Accuracy on ResNet-18 Network.

	Fed-SGD	Fed-AMS	Adp-Fed	**FED-LAMB**
CIFAR-10	90.75 ± 0.48	90.93 ± 0.22	91.57 ± 0.38	92.44 ± 0.53
TinyImageNet	67.58 ± 0.21	64.86 ± 0.83	74.17 ± 0.43	76.00 ± 0.26

6. Conclusion

We study a doubly adaptive method in the particular framework of federated learning. Built upon the success of periodic averaging, and of state-of-the-art adaptive gradient methods for single server nonconvex stochastic optimization, we derive FED-LAMB, a distributed AMSGrad method that performs local updates on each worker and periodically averages local models. When the trained model is a deep neural network, a core component of our method, FED-LAMB, is a *layer-wise* update of each local model. The main contribution of our paper is thus a federated learning optimization algorithm that leverages a double level of adaptivity: the first one stemming from a *dimension-wise* adaptivity of adaptive gradient methods, extended to their distributed (and local) counterpart, the second one is due to a *layer-wise* adaptivity making use of the particular compositionality of the considered model. Proved convergence guarantees of our scheme are provided in our contribution, and exhibit a sublinear dependence on the total number of communications rounds, and a linear speedup against the number of clients. Extensive experiments on various datasets and models, under both iid and non-iid data settings, validates that FED-LAMB is able to provide faster convergence which in turn could lead to reduced commu-

nication cost. Moreover, in many cases, FED-LAMB can
also improve the overall performance of federated learning
over prior methods.

References

[1] Alistarh, D., Grubic, D., Li, J., Tomioka, R., and Vojnovic, M. QSGD: communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1709–1720, Long Beach, CA, 2017.

[2] Chen, X., Li, X., and Li, P. Toward communication efficient adaptive gradient method. In *Proceedings of the ACM-IMS Foundations of Data Science Conference (FODS)*, pp. 119–128, Virtual Event, USA, 2020.

[3] Deng, J., Dong, W., Socher, R., Li, L., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, Miami, FL, 2009.

[4] Dozat, T. Incorporating nesterov momentum into Adam. In *Proceedings of the 4th International Conference on Learning Representations (ICLR Workshop)*, San Juan, Puerto Rico, 2016.

[5] Duchi, J. C., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, 2011.

[6] Ghadimi, S. and Lan, G. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM J. Optim.*, 23(4):2341–2368, 2013.

[7] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Las Vegas, NV, 2016.

[8] Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. Scaffold: Stochastic controlled averaging for on-device federated learning. *arXiv preprint arXiv:1910.06378*, 2019.

[9] Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015.

[10] Konečný, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[11] Krizhevsky, A. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[12] LeCun, Y. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[13] Li, M., Andersen, D. G., Park, J. W., Smola, A. J., Ahmed, A., Josifovski, V., Long, J., Shekita, E. J., and Su, B. Scaling distributed machine learning with the parameter server. In *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 583–598, Broomfield, CO, 2014.

[14] Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Process. Mag.*, 37(3):50–60, 2020.

[15] Liang, X., Shen, S., Liu, J., Pan, Z., Chen, E., and Cheng, Y. Variance reduced local sgd with lower communication complexity. *arXiv preprint arXiv:1912.12844*, 2019.

[16] Lin, Y., Han, S., Mao, H., Wang, Y., and Dally, B. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[17] McMahan, B. and Streeter, M. J. Adaptive bound optimization for online convex optimization. In *Proceedings of the 23rd Conference on Learning Theory (COLT)*, pp. 244–256, Haifa, Israel, 2010.

[18] McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1273–1282, Fort Lauderdale, FL, 2017.

[19] Nesterov, Y. Introductory lectures on convex optimization: A basic course. *Springer*, 2004.

[20] Polyak, B. T. Some methods of speeding up the convergence of iteration methods. *Mathematics and Mathematical Physics*, 1964.

[21] Recht, B., Ré, C., Wright, S. J., and Niu, F. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 693–701, Granada, Spain, 2011.

[22] Reddi, S. J., Kale, S., and Kumar, S. On the convergence of adam and beyond. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[23] Reddi, S. J., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečný, J., Kumar, S., and McMahan, H. B. Adaptive federated optimization. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, Virtual Event, Austria, 2021.

[24] Rothchild, D., Panda, A., Ullah, E., Ivkin, N., Stoica, I., Braverman, V., Gonzalez, J., and Arora, R. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, pp. 8253–8265. PMLR, 2020.

[25] Stich, S. U. Local SGD converges fast and communicates little. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, New Orleans, LA, 2019.

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

[27] Wangni, J., Wang, J., Liu, J., and Zhang, T. Gradient sparsification for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1306–1316, Montréal, Canada, 2018.

[28] Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[29] You, Y., Li, J., Reddi, S. J., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., Keutzer, K., and Hsieh, C. Large batch optimization for deep learning: Training BERT in 76 minutes. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia, 2020.

[30] Yu, H., Jin, R., and Yang, S. On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pp. 7184–7193, Long Beach, CA, 2019.

[31] Zeiler, M. D. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[32] Zhao, W., Xie, D., Jia, R., Qian, Y., Ding, R., Sun, M., and Li, P. Distributed hierarchical GPU parameter server for massive scale deep learning ads systems. In *Proceedings of Machine Learning and Systems (MLSys)*, Austin, TX, 2020.

[33] Zhou, D., Chen, J., Cao, Y., Tang, Y., Yang, Z., and Gu, Q. On the convergence of adaptive gradient methods for nonconvex optimization. *arXiv preprint arXiv:1808.05671*, 2018.

[34] Zhou, F. and Cong, G. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3219–3227, Stockholm, Sweden, 2018.

[35] Zhou, Y., Karimi, B., Yu, J., Xu, Z., and Li, P. Towards better generalization of adaptive gradient methods. In *Advances in Neural Information Processing Systems (NeurIPS)*, virtual, 2020.

A. Hyper-parameter Tuning and Algorithms

A.1. The Adp-Fed Algorithm [23]

The Adp-Fed (Adaptive Federated Optimization) is one of the baseline methods compared with FED-LAMB in our paper. The algorithm is given in Algorithm 2. The key difference between Adp-Fed and Fed-AMS [2] is that, in Adp-Fed, each local worker runs local SGD (Line 8), and an Adam optimizer is maintained for the global adaptive optimization (Line 15). In the Fed-AMS framework (as well as our FED-LAMB), each clients runs local (adaptive) AMSGrad method, and the global model is simply obtained by averaging the local models.

Algorithm 2 Adaptive Federated Optimization [23]

1: **Input**: parameter $0 < \beta_1, \beta_2 < 1$, and learning rate α_t, weight decaying parameter $\lambda \in [0, 1]$.
2: **Initialize**: $\theta_{0,i} \in \Theta \subseteq \mathbb{R}^d$, $m_0 = 0$, $v_0 = 0$, $\forall i \in [\![n]\!]$, and $\theta_0 = \frac{1}{n} \sum_{i=1}^{n} \theta_{0,i}$.

3: **for** $r = 1, \ldots, R$ **do**
4:     **parallel for device** i **do**:
5:         Set $\theta_{r,i}^0 = \theta_{r-1}$.
6:         **for** $t = 1, \ldots, T$ **do**
7:             Compute stochastic gradient $g_{r,i}^t$ at $\theta_{r,i}^0$.
8:             $\theta_{r,i}^t = \theta_{r,i}^{t-1} - \eta_l g_{r,i}^t$
9:         **end for**
10:        Devices send $\triangle_{r,i} = \theta_{r,i}^T - \theta_{r,i}^0$ to server.
11:     **end for**
12:     Server computes $\bar{\triangle}_r = \frac{1}{n} \sum_{i=1}^{n} \triangle_{r,i}$
13:     $m_r = \beta_1 m_{r-1} + (1 - \beta_1)\bar{\triangle}_r$
14:     $v_r = \beta_2 v_{r-1} + (1 - \beta_2)\bar{\triangle}_r^2$
15:     $\theta_r = \theta_{r-1} + \eta_g \frac{m_r}{\sqrt{v_r} + \epsilon}$
16: **end for**
17: **Output**: Global model parameter θ_R.

A.2. Hyper-parameter Tuning

In our empirical study, we tune the learning rate of each algorithm carefully such that the best performance is achieved. The search grids in all our experiments are provided in Table 2.

Table 2: Search grids of the learning rate.

	Learning rate range
Fed-SGD	$[0.001, 0.003, 0.005, 0.01, 0.03, 0.05, 0.1, 0.3, 0.5]$
Fed-AMS	$[0.0001, 0.0003, 0.0005, 0.001, 0.003, 0.005, 0.01, 0.03, 0.05, 0.1]$
FED-LAMB	$[0.001, 0.003, 0.005, 0.01, 0.03, 0.05, 0.1, 0.3, 0.5]$
Adp-Fed	Local η_l: $[0.0001, 0.0003, 0.0005, 0.001, 0.003, 0.005, 0.01, 0.03, 0.05, 0.1, 0.3, 0.5]$ Global η_g: $[0.0001, 0.0003, 0.0005, 0.001, 0.003, 0.005, 0.01, 0.03, 0.05, 0.1]$

B. Theoretical Analysis

We first recall in Table 3 some important notations that will be used in our following analysis.

R, T	$:=$	Number of communications rounds and local iterations (resp.)
n, D, i	$:=$	Total number of clients, portion sampled uniformly and client index
h, ℓ	$:=$	Total number of layers in the DNN and its index
$\phi(\cdot)$	$:=$	Scaling factor in FED-LAMBupdate
$\overline{\theta}$	$:=$	Global model (after periodic averaging)
$p_{r,i}^t$	$:=$	ratio computed at round r, local iteration t and for device i. $p_{r,i}^{\ell,t}$ denotes its component at layer ℓ

Table 3: Summary of notations used in the paper.

We now provide the proofs for the theoretical results of the main paper, including the intermediary Lemmas and the main convergence result, Theorem 1.

B.1. Intermediary Lemmas

Lemma. *Consider $\{\overline{\theta_r}\}_{r>0}$, the sequence of parameters obtained running Algorithm 1. Then for $i \in [\![n]\!]$:*

$$\|\overline{\theta_r} - \theta_{r,i}\|^2 \leq \alpha^2 M^2 \phi_M^2 \frac{(1-\beta_2)p}{v_0} \,,$$

where ϕ_M is defined in H4 and p is the total number of dimensions $p = \sum_{\ell=1}^h p_\ell$.

Proof. Assuming the simplest case when $T = 1$, i.e., one local iteration, then by construction of Algorithm 1, we have for all $\ell \in [\![h]\!], i \in [\![n]\!]$ and $r > 0$:

$$\theta_{r,i}^\ell = \overline{\theta_r}^\ell - \alpha\phi(\|\theta_{r,i}^{\ell,t-1}\|)p_{r,i}^j/\|p_{r,i}^\ell\| = \overline{\theta_r}^\ell - \alpha\phi(\|\theta_{r,i}^{\ell,t-1}\|)\frac{m_{r,i}^t}{\sqrt{v_r^t}}\frac{1}{\|p_{r,i}^\ell\|}$$

leading to

$$\|\overline{\theta_r} - \theta_{r,i}\|^2 = \sum_{\ell=1}^h \left\langle \overline{\theta_r}^\ell - \theta_{r,i}^\ell \mid \overline{\theta_r}^\ell - \theta_{r,i}^\ell \right\rangle$$

$$\leq \alpha^2 M^2 \phi_M^2 \frac{(1-\beta_2)p}{v_0} \,,$$

which concludes the proof. $\square$

Lemma. *Consider $\{\overline{\theta_r}\}_{r>0}$, the sequence of parameters obtained running Algorithm 1. Then for $r > 0$:*

$$\left\| \frac{\overline{\nabla}f(\theta_r)}{\sqrt{v_r^t}} \right\|^2 \geq \frac{1}{2}\left\| \frac{\nabla f(\overline{\theta_r})}{\sqrt{v_r^t}} \right\|^2 - \overline{L}\alpha^2 M^2 \phi_M^2 \frac{(1-\beta_2)p}{v_0} \,,$$

where M is defined in H2, p is the total number of dimensions $p = \sum_{\ell=1}^h p_\ell$ and ϕ_M is defined in H4.

Proof. Consider the following sequence:

$$\left\| \frac{\overline{\nabla} f(\theta_r)}{\sqrt{v_r^t}} \right\|^2 \geq \frac{1}{2} \left\| \frac{\nabla f(\overline{\theta}_r)}{\sqrt{v_r^t}} \right\|^2 - \left\| \frac{\overline{\nabla} f(\theta_r) - \nabla f(\overline{\theta}_r)}{\sqrt{v_r^t}} \right\|^2 ,$$

where the inequality is due to the Cauchy-Schwartz inequality.

Under the smoothness assumption H1 and using Lemma 1, we have

$$\left\| \frac{\overline{\nabla} f(\theta_r)}{\sqrt{v_r^t}} \right\|^2 \geq \frac{1}{2} \left\| \frac{\nabla f(\overline{\theta}_r)}{\sqrt{v_r^t}} \right\| - \left\| \frac{\overline{\nabla} f(\theta_r) - \nabla f(\overline{\theta}_r)}{\sqrt{v_r^t}} \right\|^2$$

$$\geq \frac{1}{2} \left\| \frac{\nabla f(\overline{\theta}_r)}{\sqrt{v_r^t}} \right\|^2 - \overline{L} \alpha^2 M^2 \phi_M^2 \frac{(1-\beta_2)p}{v_0} ,$$

which concludes the proof. □

B.2. Proof of Theorem 1

We now develop a proof for the two intermediary lemmas, Lemma 1 and Lemma 2, in the case when each local model is obtained after more than one local update. Then the two quantities, either the gap between the periodically averaged parameter and each local update, i.e., $\|\overline{\theta}_r - \theta_{r,i}\|^2$, and the ratio of the average gradient, more particularly its relation to the gradient of the average global model (i.e., $\left\| \frac{\overline{\nabla} f(\theta_r)}{\sqrt{v_r^t}} \right\|$ and $\left\| \frac{\nabla f(\overline{\theta}_r)}{\sqrt{v_r^t}} \right\|$), are impacted.

Theorem. *Assume H1-H4. Consider $\{\overline{\theta}_r\}_{r>0}$, the sequence of parameters obtained running Algorithm 1 with a decreasing learning rate α. Let the number of local epochs be $T \geq 1$ and $\lambda = 0$. Then, at iteration τ, we have*

$$\frac{1}{\tau} \sum_{t=1}^{\tau} \mathbb{E} \left[\left\| \frac{\nabla f(\overline{\theta}_t)}{\hat{v}_t^{1/4}} \right\|^2 \right] \leq \sqrt{\frac{M^2 p}{n}} \frac{\mathbb{E}[f(\overline{\theta}_1)] - \min_{\theta \in \Theta} f(\theta)}{\mathsf{h} \alpha_r \tau} + \frac{\phi_M \sigma^2}{\tau n} \sqrt{\frac{1-\beta_2}{M^2 p}}$$

$$+ 4\alpha \left[\frac{\alpha^2 L}{\sqrt{v_0}} M^2 (T-1)^2 \phi_M^2 (1-\beta_2) p + \frac{M^2}{\sqrt{v_0}} + \phi_M^2 \sqrt{M^2 + p\sigma^2} + \phi_M \frac{\mathsf{h}\sigma^2}{\sqrt{n}} \right] + cst.$$

Proof. Using H1, we have

$$f(\overline{\vartheta}_{r+1}) \leq f(\overline{\vartheta}_r) + \left\langle \nabla f(\overline{\vartheta}_r) \mid \overline{\vartheta}_{r+1} - \overline{\vartheta}_r \right\rangle + \sum_{\ell=1}^{L} \frac{L_\ell}{2} \|\overline{\vartheta}_{r+1}^\ell - \overline{\vartheta}_r^\ell\|^2$$

$$\leq f(\overline{\vartheta}_r) + \sum_{\ell=1}^{\mathsf{h}} \sum_{j=1}^{p_\ell} \nabla_\ell f(\overline{\vartheta}_r)^j (\overline{\vartheta}_{r+1}^{\ell,j} - \overline{\vartheta}_r^{\ell,j}) + \sum_{\ell=1}^{L} \frac{L_\ell}{2} \|\overline{\vartheta}_{r+1}^\ell - \overline{\vartheta}_r^\ell\|^2 .$$

Taking expectations on both sides leads to

$$-\mathbb{E}[\langle \nabla f(\overline{\vartheta}_r) \mid \overline{\vartheta}_{r+1} - \overline{\vartheta}_r \rangle] \leq \mathbb{E}[f(\overline{\vartheta}_r) - f(\overline{\vartheta}_{r+1})] + \sum_{\ell=1}^{L} \frac{L_\ell}{2} \mathbb{E}[\|\overline{\vartheta}_{r+1}^\ell - \overline{\vartheta}_r^\ell\|^2] . \tag{6}$$

Yet, we observe that, using the classical intermediate quantity, used for proving convergence results of adaptive optimization methods, see for instance Reddi et al. [22], we have

$$\overline{\vartheta}_r = \overline{\theta}_r + \frac{\beta_1}{1-\beta_1} (\overline{\theta}_r - \overline{\theta}_{r-1}) , \tag{7}$$

where $\bar{\theta}_r$ denotes the average of the local models at round r. Then for each layer ℓ,

$$\bar{\vartheta}_{r+1}^\ell - \bar{\vartheta}_r^\ell = \frac{1}{1-\beta_1}(\bar{\theta}_{r+1}^\ell - \bar{\theta}_r^\ell) - \frac{\beta_1}{1-\beta_1}(\bar{\theta}_r^\ell - \bar{\theta}_{r-1}^\ell)$$

$$= \frac{\alpha_r}{1-\beta_1}\frac{1}{n}\sum_{i=1}^n \frac{\phi(\|\theta_{r,i}^\ell\|)}{\|p_{r,i}^\ell\|}p_{r,i}^\ell - \frac{\alpha_{r-1}}{1-\beta_1}\frac{1}{n}\sum_{i=1}^n \frac{\phi(\|\theta_{r-1,i}^\ell\|)}{\|p_{r-1,i}^\ell\|}p_{r-1,i}^\ell$$

$$= \frac{\alpha\beta_1}{1-\beta_1}\frac{1}{n}\sum_{i=1}^n \left(\frac{\phi(\|\theta_{r,i}^\ell\|)}{\sqrt{v_r^t}\|p_{r,i}^\ell\|} - \frac{\phi(\|\theta_{r-1,i}^\ell\|)}{\sqrt{v_{r-1}^t}\|p_{r-1,i}^\ell\|}\right)m_{r-1}^t + \frac{\alpha}{n}\sum_{i=1}^n \frac{\phi(\|\theta_{r,i}^\ell\|)}{\sqrt{v_r^t}\|p_{r,i}^\ell\|}g_{r,i}^t, \tag{8}$$

where we have assumed a constant learning rate α.

We note for all $\theta \in \Theta$, the majorant $G > 0$ such that $\phi(\|\theta\|) \le G$. Then, following (6), we obtain

$$-\mathbb{E}[\langle \nabla f(\bar{\vartheta}_r) \,|\, \bar{\vartheta}_{r+1} - \bar{\vartheta}_r \rangle] \le \mathbb{E}[f(\bar{\vartheta}_r) - f(\bar{\vartheta}_{r+1})] + \sum_{\ell=1}^L \frac{L_\ell}{2}\mathbb{E}[\|\bar{\vartheta}_{r+1} - \bar{\vartheta}_r\|^2]. \tag{9}$$

Developing the LHS of (9) using (8) leads to

$$\langle \nabla f(\bar{\vartheta}_r) \,|\, \bar{\vartheta}_{r+1} - \bar{\vartheta}_r \rangle = \sum_{\ell=1}^h \sum_{j=1}^{p_\ell} \nabla_\ell f(\bar{\vartheta}_r)^j (\bar{\vartheta}_{r+1}^{\ell,j} - \bar{\vartheta}_r^{\ell,j})$$

$$= \frac{\alpha\beta_1}{1-\beta_1}\frac{1}{n}\sum_{\ell=1}^h \sum_{j=1}^{p_\ell} \nabla_\ell f(\bar{\vartheta}_r)^j \left[\sum_{i=1}^n \left(\frac{\phi(\|\theta_{r,i}^\ell\|)}{\sqrt{v_r^t}\|p_{r,i}^\ell\|} - \frac{\phi(\|\theta_{r-1,i}^\ell\|)}{\sqrt{v_{r-1}^t}\|p_{r-1,i}^\ell\|}\right)m_{r-1}^t\right]$$

$$\underbrace{-\frac{\alpha}{n}\sum_{\ell=1}^h \sum_{j=1}^{p_\ell} \nabla_\ell f(\bar{\vartheta}_r)^j \sum_{i=1}^n \frac{\phi(\|\theta_{r,i}^\ell\|)}{\sqrt{v_r^t}\|p_{r,i}^\ell\|}g_{r,i}^{t,l,j}}_{=A_1}. \tag{10}$$

We change all index r to iteration t. Suppose T is the number of local iterations. We can write (10) as

$$A_1 = -\alpha_t \langle \nabla f(\bar{\vartheta}_t), \frac{\bar{g}_t}{\sqrt{\hat{v}_t}}\rangle,$$

where $\bar{g}_t = \frac{1}{n}\sum_{i=1}^n \bar{g}_{t,i}$, with $\bar{g}_{t,i} = \left[\frac{\phi(\|\theta_{t,i}^1\|)}{\|p_{t,i}^1\|}g_{t,i}^1, ..., \frac{\phi(\|\theta_{t,i}^L\|)}{\|p_{t,i}^L\|}g_{t,i}^L\right]$ representing the normalized gradient (concatenated by layers) of the i-th device. It holds that

$$\langle \nabla f(\bar{\vartheta}_t), \frac{\bar{g}_t}{\sqrt{\hat{v}_t}}\rangle = \frac{1}{2}\|\frac{\nabla f(\bar{\vartheta}_t)}{\hat{v}_t^{1/4}}\|^2 + \frac{1}{2}\|\frac{\bar{g}_t}{\hat{v}_t^{1/4}}\|^2 - \|\frac{\nabla f(\bar{\vartheta}_t) - \bar{g}_t}{\hat{v}_t^{1/4}}\|^2. \tag{11}$$

To bound the last term on the RHS, we have

$$\|\frac{\nabla f(\bar{\vartheta}_t) - \bar{g}_t}{\hat{v}_t^{1/4}}\|^2 = \|\frac{\frac{1}{n}\sum_{i=1}^n(\nabla f(\bar{\vartheta}_t) - \bar{g}_{t,i})}{\hat{v}_t^{1/4}}\|^2$$

$$\leq \frac{1}{n}\sum_{i=1}^n \|\frac{\nabla f(\bar{\vartheta}_t) - \bar{g}_{t,i}}{\hat{v}_t^{1/4}}\|^2$$

$$\leq \frac{2}{n}\sum_{i=1}^n \Big(\|\frac{\nabla f(\bar{\vartheta}_t) - \nabla f(\bar{\theta}_t)}{\hat{v}_t^{1/4}}\|^2 + \|\frac{\nabla f(\bar{\theta}_t) - \bar{g}_{t,i}}{\hat{v}_t^{1/4}}\|^2\Big).$$

By Lipschitz smoothness of the loss function, the first term admits

$$\frac{2}{n}\sum_{i=1}^n \|\frac{\nabla f_i(\bar{\vartheta}_t) - \nabla f_i(\bar{\theta}_t)}{\hat{v}_t^{1/4}}\|^2 \leq \frac{2}{n\sqrt{v_0}}\sum_{i=1}^n L_\ell\|\bar{\vartheta}_t - \bar{\theta}_t\|^2$$

$$= \frac{2L_\ell}{n\sqrt{v_0}}\frac{\beta_1^2}{(1-\beta_1)^2}\sum_{i=1}^n \|\bar{\theta}_t - \bar{\theta}_{t-1}\|^2$$

$$\leq \frac{2\alpha_r^2 L_\ell}{n\sqrt{v_0}}\frac{\beta_1^2}{(1-\beta_1)^2}\sum_{l=1}^L\sum_{i=1}^n \|\frac{\phi(\|\theta_{t,i}^l\|)}{\|p_{t,i}^l\|}p_{t,i}^l\|^2$$

$$\leq \frac{2\alpha_r^2 L_\ell p\phi_M^2}{\sqrt{v_0}}\frac{\beta_1^2}{(1-\beta_1)^2}.$$

For the second term,

$$\frac{2}{n}\sum_{i=1}^n \|\frac{\nabla f(\bar{\theta}_t) - \bar{g}_{t,i}}{\hat{v}_t^{1/4}}\|^2 \leq \frac{4}{n}\Big(\underbrace{\sum_{i=1}^n \|\frac{\nabla f(\bar{\theta}_t) - \nabla f(\theta_{t,i})}{\hat{v}_t^{1/4}}\|^2}_{B_1} + \underbrace{\sum_{i=1}^n \|\frac{\nabla f(\theta_{t,i}) - \bar{g}_{t,i}}{\hat{v}_t^{1/4}}\|^2}_{B_2}\Big). \tag{12}$$

Using the smoothness of f_i we can transform B_1 into consensus error by

$$B_1 \leq \frac{L}{\sqrt{v_0}}\sum_{i=1}^n \|\bar{\theta}_t - \theta_{t,i}\|^2$$

$$= \frac{\alpha_r^2 L}{\sqrt{v_0}}\sum_{i=1}^n\sum_{l=1}^L \|\sum_{j=\lfloor t\rfloor_T+1}^t \Big(\frac{\phi(\|\theta_{j,i}^l\|)}{\|p_{j,i}^l\|}p_{j,i}^l - \frac{1}{n}\sum_{k=1}^n\frac{\phi(\|\theta_{j,k}^l\|)}{\|p_{j,k}^l\|}p_{j,k}^l\Big)\|^2 \tag{13}$$

$$\leq n\frac{\alpha_t^2 L}{\sqrt{v_0}}M^2(T-1)^2\phi_M^2(1-\beta_2)p,$$

where the last inequality stems from Lemma 1 in the particular case where $\theta_{t,i}$ are averaged every $ct+1$ local iterations for any integer c, since $(t-1) - (\lfloor t\rfloor_T + 1) + 1 \leq T - 1$.

We now develop the expectation of B_2 under the simplification that $\beta_1 = 0$:

$$\mathbb{E}[B_2] = \mathbb{E}[\sum_{i=1}^{n} \|\frac{\nabla f(\theta_{t,i}) - \bar{g}_{t,i}}{\hat{v}_t^{1/4}}\|^2]$$

$$\leq \frac{nM^2}{\sqrt{v_0}} + n\phi_M^2 \sqrt{M^2 + p\sigma^2} - 2\sum_{i=1}^{n} \mathbb{E}[\langle \nabla f(\theta_{t,i}), \bar{g}_{t,i}\rangle / \sqrt{\hat{v}_t}]$$

$$= \frac{nM^2}{\sqrt{v_0}} + n\phi_M^2 \sqrt{M^2 + p\sigma^2} - 2\sum_{i=1}^{n} \sum_{\ell=1}^{L} \mathbb{E}[\langle \nabla_\ell f(\theta_{t,i}), \frac{\phi(\|\theta_{t,i}^l\|)}{\|p_{t,i}^l\|} g_{t,i}^l \rangle / \sqrt{\hat{v}_t^l}]$$

$$= \frac{nM^2}{\sqrt{v_0}} + n\phi_M^2 \sqrt{M^2 + p\sigma^2} - 2\sum_{i=1}^{n} \sum_{l=1}^{L} \sum_{i=1}^{p_l} \mathbb{E}[\nabla_l f(\theta_{t,i})^j \frac{\phi(\|\theta_{t,i}^{l,j}\|)}{\sqrt{\hat{v}_t^{l,j}} \|p_{t,i}^{l,j}\|} g_{t,i}^{l,j}]$$

$$\leq \frac{nM^2}{\sqrt{v_0}} + n\phi_M^2 \sqrt{M^2 + p\sigma^2} - 2\sum_{i=1}^{n} \sum_{l=1}^{L} \sum_{i=1}^{p_l} \mathbb{E}\left[\sqrt{\frac{1-\beta_2}{M^2 p_\ell}} \phi(\|\theta_{r,i}^{l,j}\|) \nabla_l f(\theta_{t,i})^j g_{t,i}^{l,j}\right]$$

$$- 2\sum_{i=1}^{n} \sum_{l=1}^{L} \sum_{j=1}^{p_l} E\left[\left(\phi(\|\theta_{r,i}^{l,j}\|) \nabla_l f(\theta_{t,i})^j \frac{g_{r,i}^{t,l,j}}{\|p_{r,i}^{l,j}\|}\right) \mathbb{1}\left(\text{sign}(\nabla_l f(\theta_{t,i})^j \neq \text{sign}(g_{r,i}^{t,l,j}))\right)\right],$$

where we use assumption H2, H3 and H4. Yet,

$$-\mathbb{E}\left[\left(\phi(\|\theta_{r,i}^{l,j}\|) \nabla_l f(\theta_{t,i})^j \frac{g_{r,i}^{t,l,j}}{\|p_{r,i}^{l,j}\|}\right) \mathbb{1}\left(\text{sign}(\nabla_l f(\theta_{t,i})^j \neq \text{sign}(g_{r,i}^{t,l,j}))\right)\right]$$

$$\leq \phi_M \nabla_l f(\theta_{t,i})^j \mathbb{P}\left[\text{sign}(\nabla_l f(\theta_{t,i})^j \neq \text{sign}(g_{r,i}^{t,l,j}))\right].$$

Then we have

$$\mathbb{E}[B_2] \leq \frac{nM^2}{\sqrt{v_0}} + n\phi_M^2 \sqrt{M^2 + p\sigma^2} - 2\phi_m \sqrt{\frac{1-\beta_2}{M^2 p}} \sum_{i=1}^{n} \mathbb{E}[\|[\nabla f(\theta_{t,i})\|^2] + \phi_M \frac{\mathsf{h}\sigma^2}{\sqrt{n}}$$

Thus, (12) becomes

$$\frac{2}{n} \sum_{i=1}^{n} \|\frac{\nabla f_i(\bar{\theta}_t) - \bar{g}_{t,i}}{\hat{v}_t^{1/4}}\|^2 \leq 4\left[\frac{\alpha_t^2 L l}{\sqrt{v_0}} \alpha_r^2 M^2 (T-1)^2 \phi_M^2 (1-\beta_2)p + \frac{M^2}{\sqrt{v_0}} + \phi_M^2 \sqrt{M^2 + p\sigma^2} + \phi_M \frac{\mathsf{h}\sigma^2}{\sqrt{n}}\right]$$

Substituting all ingredients into (11), we obtain

$$-\alpha_t \mathbb{E}[\langle \nabla f(\bar{\vartheta}_t), \frac{\bar{g}_t}{\sqrt{\hat{v}_t}}\rangle] \leq -\frac{\alpha_t}{2} \mathbb{E}[\|\frac{\nabla f(\bar{\vartheta}_t)}{\hat{v}_t^{1/4}}\|^2] - \frac{\alpha_t}{2} \mathbb{E}[\|\frac{\bar{g}_t}{\hat{v}_t^{1/4}}\|^2] + \frac{2\alpha_t^3 L_\ell p \phi_M^2}{\sqrt{v_0}} \frac{\beta_1^2}{(1-\beta_1)^2}$$

$$+ 4\left[\frac{\alpha_t^2 L}{\sqrt{v_0}} M^2 (T-1)^2 \phi_M^2 (1-\beta_2)p + \frac{M^2}{\sqrt{v_0}} + \phi_M^2 \sqrt{M^2 + p\sigma^2} + \phi_M \frac{\mathsf{h}\sigma^2}{\sqrt{n}}\right].$$

At the same time, we have

$$\mathbb{E}[\|\frac{\bar{g}_t}{\hat{v}_t^{1/4}}\|^2] = \frac{1}{n^2} \mathbb{E}[\|\frac{\sum_{i=1}^{n} \bar{g}_{t,i}}{\hat{v}_t^{1/4}}\|^2]$$

$$= \frac{1}{n^2} \mathbb{E}[\sum_{l=1}^{L} \sum_{i=1}^{n} \|\frac{\phi(\|\theta_{t,i}^l\|)}{\hat{v}^{1/4} \|p_{t,i}^l\|} g_{t,i}^l\|^2]$$

$$\geq \phi_m^2 (1-\beta_2) \mathbb{E}\left[\|\frac{1}{n} \sum_{i=1}^{n} \frac{\nabla f(\theta_{t,i})}{\hat{v}^{1/4}}\|^2\right]$$

$$= \phi_m^2 (1-\beta_2) \mathbb{E}\left[\|\frac{\overline{\nabla} f(\theta_t)}{\hat{v}^{1/4}}\|^2\right].$$

Regarding $\left\| \frac{\overline{\nabla} f(\theta_t)}{\hat{v}_t^{1/4}} \right\|^2$, we have

$$\left\| \frac{\overline{\nabla} f(\theta_t)}{\hat{v}_t^{1/4}} \right\|^2 \geq \frac{1}{2} \left\| \frac{\nabla f(\overline{\theta}_t)}{\hat{v}_t^{1/4}} \right\|^2 - \left\| \frac{\overline{\nabla} f(\theta_t) - \nabla f(\overline{\theta}_t)}{\hat{v}_t^{1/4}} \right\|^2$$

$$\geq \frac{1}{2} \left\| \frac{\nabla f(\overline{\theta}_t)}{\hat{v}_t^{1/4}} \right\|^2 - \left\| \frac{\frac{1}{n} \sum_{i=1}^{n} (\nabla f(\theta_{t,i}) - \nabla f(\bar{\theta}_i))}{\hat{v}_t^{1/4}} \right\|^2$$

$$\geq \frac{1}{2} \left\| \frac{\nabla f(\overline{\theta}_t)}{\hat{v}_t^{1/4}} \right\|^2 - \frac{\alpha_t^2 L_\ell}{\sqrt{v_0}} M^2 (T-1)^2 \phi_M^2 (1-\beta_2) p,$$

where the last line is due to (13). Therefore, we have obtained

$$A_1 \leq -\frac{\phi_m^2(1-\beta_2)}{2} \left\| \frac{\nabla f(\overline{\theta}_t)}{\hat{v}_t^{1/4}} \right\|^2 + \frac{\alpha_r^2 L_\ell}{\sqrt{v_0}} M^2 (T-1)^2 \phi_m^2 \phi_M^2 (1-\beta_2)^2 p + \frac{2\alpha^3 L_\ell p \phi_M^2}{\sqrt{v_0}} \frac{\beta_1^2}{(1-\beta_1)^2}$$

$$+ 4\alpha_t \left[\frac{\alpha_t^2 L}{\sqrt{v_0}} M^2 (T-1)^2 \phi_M^2 (1-\beta_2) p + \frac{M^2}{\sqrt{v_0}} + \phi_M^2 \sqrt{M^2 + p\sigma^2} + \phi_M \frac{h\sigma^2}{\sqrt{n}} \right].$$

Substitute back into (10), and leave other derivations unchanged. Assuming $M \leq 1$, we have the following

$$\frac{1}{\tau} \sum_{t=1}^{\tau} \mathbb{E} \left[\left\| \frac{\nabla f(\overline{\theta}_t)}{\hat{v}_t^{1/4}} \right\|^2 \right]$$

$$\lesssim \sqrt{\frac{M^2 p}{n}} \frac{f(\bar{\vartheta}_1) - \mathbb{E}[f(\bar{\vartheta}_{\tau+1})]}{h\alpha_t \tau} + \frac{\alpha_t}{n^2} \sum_{r=1}^{\tau} \sum_{i=1}^{n} \sigma_i^2 \mathbb{E} \left[\left\| \frac{\phi(\|\theta_{r,i}^\ell\|)}{\sqrt{v_t} \|p_{r,i}^\ell\|} \right\|^2 \right] + \frac{2\alpha^3 L_\ell p \phi_M^2}{\sqrt{v_0}} \frac{\beta_1^2}{(1-\beta_1)^2}$$

$$+ 4\alpha_t \left[\frac{\alpha_t^2 L_\ell}{\sqrt{v_0}} M^2 (T-1)^2 \phi_M^2 (1-\beta_2) p + \frac{M^2}{\sqrt{v_0}} + \phi_M^2 \sqrt{M^2 + p\sigma^2} + \phi_M \frac{h\sigma^2}{\sqrt{n}} \right] + \frac{\overline{L}\beta_1^2 h(1-\beta_2) M^2 \phi_M^2 n}{2(1-\beta_1)^2 v_0}$$

$$+ \frac{\alpha_t \beta_1}{1-\beta_1} \sqrt{(1-\beta_2) p} \frac{h M^2}{\sqrt{v_0}} + \overline{L} \alpha_t^2 M^2 \phi_M^2 \frac{(1-\beta_2) p}{T v_0}$$

$$\leq \sqrt{\frac{M^2 p}{n}} \frac{\mathbb{E}[f(\bar{\theta}_1)] - \min_{\theta \in \Theta} f(\theta)}{h\alpha_t \tau} + \frac{\phi_M \sigma^2}{\tau n} \sqrt{\frac{1-\beta_2}{M^2 p}}$$

$$+ 4\alpha_t \left[\frac{\alpha_t^2 L_\ell}{\sqrt{v_0}} M^2 (T-1)^2 \phi_M^2 (1-\beta_2) p + \frac{M^2}{\sqrt{v_0}} + \phi_M^2 \sqrt{M^2 + p\sigma^2} + \phi_M \frac{h\sigma^2}{\sqrt{n}} \right]$$

$$+ \frac{\alpha_t \beta_1}{1-\beta_1} \sqrt{(1-\beta_2) p} \frac{h M^2}{\sqrt{v_0}} + \overline{L} \alpha_t^2 M^2 \phi_M^2 \frac{(1-\beta_2) p}{T v_0} + \frac{\overline{L}\beta_1^2 h(1-\beta_2) M^2 \phi_M^2 n}{2(1-\beta_1)^2 v_0} + \frac{2\alpha^3 L_\ell p \phi_M^2}{\sqrt{v_0}} \frac{\beta_1^2}{(1-\beta_1)^2}.$$

And if we set the learning rate to be of order $\mathcal{O}(\frac{1}{L\sqrt{\tau}})$ then:

$$\frac{1}{\tau} \sum_{t=1}^{\tau} \mathbb{E} \left[\left\| \frac{\nabla f(\overline{\theta}_t)}{\hat{v}_t^{1/4}} \right\|^2 \right] \leq \mathcal{O} \left(\sqrt{\frac{M^2 p}{n}} \frac{1}{\sqrt{h\tau}} + \frac{\sigma^2}{\tau n \sqrt{p}} + \frac{(T-1)^2 p}{\tau^{3/2} L^3} \right).$$

This concludes the proof.

□