Momentum Approximation in Asynchronous Private Federated Learning

Tao Yu[*] Amazon
Congzheng Song Apple
Jianyu Wang Apple
Mona Chinits Apple

Abstract

Asynchronous protocols have been shown to improve the scalability of federated learning (FL) with a massive number of clients. Meanwhile, momentum-based methods can achieve the best model quality in synchronous FL. However, naively applying momentum in asynchronous FL algorithms leads to slower convergence and degraded model performance. It is still unclear how to effective combinie these two techniques together to achieve a win-win. In this paper, we find that asynchrony introduces implicit bias to momentum updates. In order to address this problem, we propose momentum approximation that minimizes the bias by finding an optimal weighted average of all historical model updates. Momentum approximation is compatible with secure aggregation as well as differential privacy, and can be easily integrated in production FL systems with a minor communication and storage cost. We empirically demonstrate that on benchmark FL datasets, momentum approximation can achieve 1.15–$4\times$ speed up in convergence compared to naively combining asynchronous FL with momentum.

1 Introduction

Practical deployment of synchronous federated learning (SyncFL) [34] encounters scalability issue due to the requirement on global synchronization of clients' model updates, wherein the central aggregation are contingent upon the completion of local training and communication across all participating clients. In order to address this issue, asynchronous FL (AsyncFL) [6, 41, 43, 49, 56, 61] was proposed, which allows concurrent model updates at both the server and the clients' side. One concrete example is FedBuff [41], which is the state-of-the-art AsyncFL method and has been deployed in many production systems [22, 51]. In each FedBuff iteration, the server first broadcasts the global model and triggers local training on K randomly sampled clients, then, receives clients' local model updates in a buffer. Once the buffer reaches a target cohort size $C \ll K$, the server will directly proceed to the next iteration without waiting for the all K clients finish computation. As a result, the buffer gets filled up much quicker than SyncFL and the latency per iteration improves significantly [14]. However, since clients sampled in all previous iterations can contribute to the current global model update via stale gradients, AsyncFL methods typically have slower model convergence w.r.t iterations than SyncFL.

On the other hand, momentum-based optimizers such as momentum SGD and Adam have become dominant in the deep learning community due to their superior performance. Similar observations also appeared in SyncFL. For example, researchers found that applying momentum methods for the server model updates (e.g., FedAvgM [21] and FedAdam [45]) can greatly improve the final model quality and convergence speed.

Given the appealing benefits of asynchrony and momentum, it is desired to combine them to achieve a win-win in both efficiency and model quality. Unfortunately, the naive combination does not

[*]Work was done while interning at Apple.

Workshop on Federated Foundation Models in Conjunction with NeurIPS 2024 (FL@FM-NeurIPS'24)

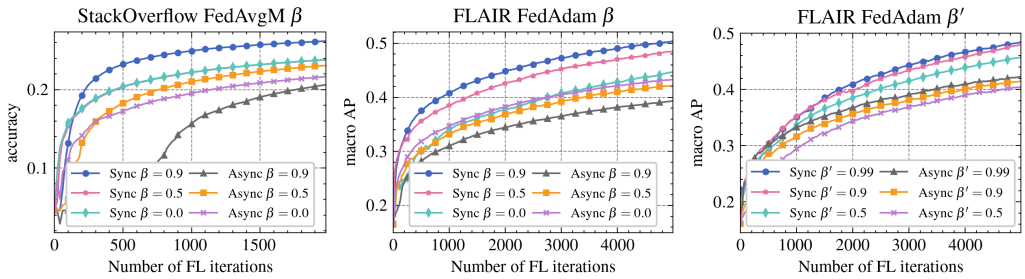

Figure 1: (Left and Middle) In SyncFL, FedAvgM and FedAdam with momentum parameter $\beta = 0.9$ converges fastest while it is not the case in AsyncFL: no momentum ($\beta = 0$) or smaller $\beta = 0.5$ is better. (Right) The parameter β' for the second moments in FedAdam, on the other hand, has consistent impact on SyncFL and AsyncFL, i.e. larger $\beta' = 0.99$ is better.

work. For instance, [39, 62] showed that sophisticated tuning of the momentum parameter β is very critical in asynchronous SGD (AsyncSGD). Rather than consistently using a large β (e.g. 0.9) in the synchronous setting, a smaller or even negative β is preferred and the best value may vary across datasets. We observe the same phenomenon for AsyncFL. As shown in Figure 1, both FedAvgM and FedAdam with $\beta = 0.9$ underperforms smaller β in asynchronous setting while the pattern is the opposite if updates are synchronous.

Prior AsyncFL works proposed to down-scale the stale client updates before aggregation to control the impact of staleness [43, 56]. However, this does not help in combining asynchrony and momentum. As shown in the experiments of FedBuff [41], even with lower weights for stale updates, β still needs to be carefully tuned on different datasets and the best value can be 0 (i.e. no momentum). It remains an open question: *is it possible to effectively integrate asynchrony and momentum in FL to simultaneously harness the advantages in scalability and better model quality?*

Contributions. In this paper, we provide an affirmative answer to the above question. Motivated by the fact that momentum method itself utilizes all past gradients by taking an exponential average (i.e., has unbounded staleness), we argue that the key issue in applying momentum to AsyncFL may not be the large staleness of model updates. Instead, the real problem is that naive asynchronous momentum method does not properly exploit the past information. We demonstrate that asynchrony introduces implicit weight bias to past gradients and hence, removes the momentum effect.

In order to address this problem, we propose a new algorithm named *momentum approximation*, which solves a least square problem in each FL iteration t to find the best coefficients to weight the historical model updates before t, such that the weighted historical updates are close to the momentum updates as in the synchronous setting and thus retain the acceleration from momentum. We highlight some key features of this algorithm:

- Momentum approximation is compatible with any momentum-based federated optimizer and its convergence pattern behaves similar to SyncFL. It resolves the need of extensively tuning β from a wider range for different tasks in prior works. One can consistently set β found in SyncFL to get the best model quality in the AsyncFL.
- Momentum approximation can be easily integrated in production FL systems, and inherits all the benefits from FedBuff, such as its scalability, robustness and compatibility to privacy. It incurs only a minor communication of a iteration number in addition to model updates, and storage cost of historical updates on the server side.
- We empirically demonstrate that on two large-scale FL benchmarks, StackOverflow [1] and FLAIR [46], momentum approximation achieves 1.15–4$\times$ speed up in convergence and 3–20% improvements in utility compared to vanilla FedBuff with momentum.

2 Background

In federated learning (FL), we aim to train a model $\boldsymbol{\theta} \in \mathbb{R}^d$ with m clients collaboratively. In iteration t of FL, a cohort of clients is sampled and the server broadcasts the current global model $\boldsymbol{\theta}_t$ to the sampled clients $\mathcal{K}_t$. Each sampled client k trains on their local dataset, and then submits the

model updates $\Delta_k(\boldsymbol{\theta}_t)$ before and after the local training back to the server. In SyncFL, the server waits for the local model updates from all $K = |\mathcal{K}_t|$ clients and uses the *aggregated model updates* $\boldsymbol{d}_t = \frac{1}{K}\sum_{k\in\mathcal{K}_t}\Delta_k(\boldsymbol{\theta}_t)$ to update the global model before proceeding to the next iteration. More formally, synchronous federated averaging (FedAvg) [34] algorithm updates the global model as $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\boldsymbol{d}_t$ where η denotes the server learning rate.

Momentum-based optimizers. In practice, momentum-based optimizers [21, 45, 53] on the server side are often more preferred than FedAvg as they can either greatly accelerate convergence or improve the final model quality given a fixed iteration budget. We denote these optimizers as SERVEROPT, and the update rule of which can be formulated as:

$$\boldsymbol{m}_t = \beta\boldsymbol{m}_{t-1} + (1-\beta)\boldsymbol{d}_t,\ \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\boldsymbol{H}_t^{-1}\boldsymbol{m}_t, \tag{1}$$

where $\beta \in [0, 1)$ is the momentum parameter and $\boldsymbol{m}_t$ is the momentum buffer. $\boldsymbol{H}_t^{-1}$ is the preconditioner where $\boldsymbol{H}_t = \boldsymbol{I}_d$ in FedAvgM [21], and $\boldsymbol{H}_t$ is the square root of accumulated or exponential moving average of $\boldsymbol{d}_t$'s second moments in adaptive optimizers such as FedAdaGrad and FedAdam [12, 26, 45].

FL with differential privacy. Though the clients' raw data is never shared with the server in FL, the model updates Δ_k can still reveal sensitive information [36, 40, 65]. Differential privacy (DP) is a standard approach to prevent leakage from Δ_k and provide a meaningful privacy guarantee.

Definition 2.1 (Differential Privacy [16])**.** A randomized algorithm $\mathcal{A} : \mathcal{D} \mapsto \mathcal{R}$ is (ϵ, δ)-differentially private, if for any pair of neighboring training populations $\mathcal{D}$ and $\mathcal{D}'$ and for any subset of outputs $\mathcal{S} \subseteq \mathcal{R}$, it holds that

$$\Pr[\mathcal{A}(\mathcal{D}) \in \mathcal{S}] \leq e^{\epsilon} \cdot \Pr[\mathcal{A}(\mathcal{D}') \in \mathcal{S}] + \delta. \tag{2}$$

We consider client-level DP where a training population $\mathcal{D}'$ is the neighbor of $\mathcal{D}$ if $\mathcal{D}'$ can be obtained by adding or removing one client from $\mathcal{D}$, and vice versa. Gaussian Mechanism [15] can be easily combined with FL [35] to enable DP, where two more additional steps are required in each iteration: (1) each client model update is clipped by $\text{clip}(\Delta_k, S) = \Delta_k \cdot \min(1, S/\|\Delta_k\|_2)$ with L^2 sensitivity bound S, and (2) the aggregated clipped model updates are added with Gaussian noise as $\sum_k \text{clip}(\Delta_k, S) + \mathcal{N}(0, \sigma^2 S^2 \boldsymbol{I})$ where σ is calibrated from a standard privacy accountant such as Rényi DP [37]. We also assume a secure aggregation protocol is used so that the server learns only the sum $\sum_k \Delta_k$ but never the individual model updates Δ_k [4, 22, 48].

3 Applying Momentum to Asynchronous FL

In this section, we first demonstrate the problem in naively combining momentum and AsyncFL, and then introduce momentum approximation to address it. Unless otherwise stated, we focus on the FedBuff algorithm, which is a general and state-of-the-art AsyncFL algorithm.

Notation and assumptions. For a matrix $\boldsymbol{A}$, we use $\boldsymbol{A}_{[i,:]}$ to denote i-th row, $\boldsymbol{A}_{[:,j]}$ the j-th column, and $\boldsymbol{A}_{[i,j]}$ the (i, j)-th entry of $\boldsymbol{A}$. We denote the staleness as $\tau(k)$ for client k, i.e. k is sampled at iteration $t - \tau(k)$ and their updates is received at t. We let $\boldsymbol{e}_i \in \{0, 1\}^T$ denote the one-hot encoding vector where all entries are 0 except for the i-th entry being 1, and $\mathbf{1} = [1, 1, \ldots, 1]^\top$ denote a vector with all ones. We denote $\mathcal{K}_t$ as the set of K clients sampled at iteration t, and $\mathcal{C}_t$ as the set of C clients whose updates received by server at iteration t.

To gain insights on the impact of momentum in AsyncFL, we define $\boldsymbol{d}_t^\star = \frac{1}{m}\sum_{k=1}^{m}\Delta_k(\boldsymbol{\theta}_t)$, i.e. the average of local model updates over all m clients starting from the same point $\boldsymbol{\theta}_t$. We make the following assumptions throughout.

Assumption 3.1 (Bounded Population Client Update)**.** For each iteration t, $\|\boldsymbol{d}_t^\star\|_2^2 \leq S^2$.

Assumption 3.2 (Bounded Global Dissimilarity)**.** For all clients $k \in [m]$ and for each iteration t, $\mathbb{E}_{k\sim[m]}\|\Delta_k(\boldsymbol{\theta}_t) - \boldsymbol{d}_t^\star\|_2^2 \leq G^2$.

Assumption 3.3. (Bounded Client Subset Sampling Error) For the sampled clients set $\mathcal{K}_t$ in each iteration t, $\mathbb{E}_{k\sim\mathcal{K}_t}\|\Delta(\boldsymbol{\theta}_t)_k - \boldsymbol{d}_t^\star\|_2^2 \leq \|\boldsymbol{d}_t - \boldsymbol{d}_t^\star\|_2^2 + \rho^2$.

Assumption 3.4. (Random Arrival Order of Sampled Clients) For each iteration t and $s \leq t$, each client $k \in \mathcal{K}_{t,s}$ is a random sample from the set $\mathcal{K}_s$.

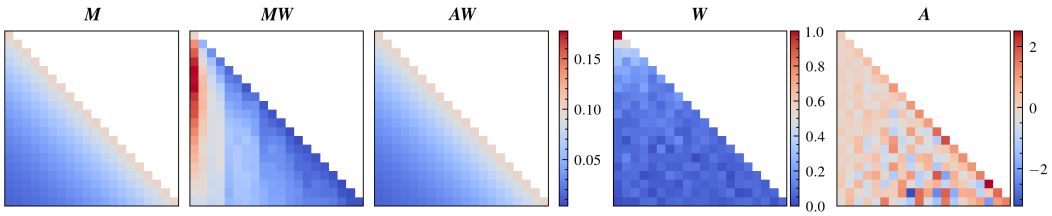

Figure 2: Visualization of the desired momentum matrix $\boldsymbol{M}$ ($\beta = 0.9$), the implicit momentum matrix $\boldsymbol{MW}$, the approximated momentum matrix $\boldsymbol{AW}$, the staleness coefficient matrix $\boldsymbol{W}$, and the solved weighting matrix $\boldsymbol{A}$ in momentum approximation.

Assumptions 3.1 and 3.2 are common in the FL literature [29, 41, 55, 60]. Assumptions 3.1 also trivially holds with DP. Assumption 3.3 is a natural extension given Assumption 3.2 and $\rho^2 \leq G^2$. Assumption 3.4 is based on the fact the timing of client participation tends to be random among sampled clients. We justify Assumption 3.4 in detail in Appendix B.2.

3.1 Implicit Momentum Bias

In order to get a better understanding on the convergence issue of AsyncFL with momentum, we first present a general update rule for FL and then compare synchronous momentum methods and asynchronous ones as special cases.

Without loss of generality, we define $\boldsymbol{r}_t \in \mathbb{R}^d$ as the aggregated pseudo-gradient (or model updates) received by the server at iteration t, and denote $\boldsymbol{R} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_T] \in \mathbb{R}^{d\times T}$. Then, the following proposition holds [11].

Proposition 3.5. *Suppose the server model $\boldsymbol{\theta}$ is updated using momentum method as follows:*

$$\boldsymbol{m}_t = \beta \boldsymbol{m}_{t-1} + (1-\beta)\boldsymbol{r}_t,\ \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \boldsymbol{m}_t.$$

This update rule is equivalent to $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta(1-\beta)\sum_{s=1}^{t}\beta^{t-s}\boldsymbol{r}_s$. The final model after total T iterations can be written as:

$$\boldsymbol{\theta}_{T+1} = \boldsymbol{\theta}_1 - \eta \boldsymbol{R}\boldsymbol{M}^\top \mathbf{1}, \tag{3}$$

where $\boldsymbol{M} \in \mathbb{R}^{T\times T}$ is a lower-triangular matrix defined as: $\boldsymbol{M}_{[t,s]} = \begin{cases} \beta^{t-s}(1-\beta) & \text{if } t >= s \\ 0 & \text{otherwise} \end{cases}$.

With the above general update rule, both SyncFL and AsyncFL can be treated as special cases. For SyncFL with all clients participating, the received (pseudo-) gradient on server is just the aggregated local model updates from all m clients, that is, $\boldsymbol{r}_t = \boldsymbol{d}_t^\star$. For SyncFL with client sub-sampling, $\boldsymbol{r}_t = \boldsymbol{d}_t$. Denote $\boldsymbol{D}^\star = [\boldsymbol{d}_1^\star, \boldsymbol{d}_2^\star, \ldots, \boldsymbol{d}_T^\star]$ and $\boldsymbol{D} = [\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_T]$, then

$$\boldsymbol{\theta}_{T+1}^\star = \boldsymbol{\theta}_1 - \eta \boldsymbol{D}^\star \boldsymbol{M}^\top \mathbf{1}, \quad \boldsymbol{\theta}_{T+1}^{\text{sync}} = \boldsymbol{\theta}_1 - \eta \boldsymbol{D}\boldsymbol{M}^\top \mathbf{1}. \tag{4}$$

For asynchronous setting, the concrete expression of $\boldsymbol{r}_t$ is more complicated. At each iteration of FedBuff, the server broadcasts the latest model $\boldsymbol{\theta}_t$ to K random sampled clients to trigger local training and applies a global update after receiving C local model updates from $\mathcal{C}_t$. However, these received local model updates can be stale. Let $\mathcal{K}_{t,s}$ be the set of clients sampled at iteration s with updates received at iteration t where $|\mathcal{K}_{t,s}| = C_{t,s} \geq 0$ and $\sum_{s=1}^{t} C_{t,s} = C$. Then, the current (pseudo-) gradient on server can be written as a weighted average of all past updates:

$$\begin{aligned}
\boldsymbol{r}_t &= \frac{1}{C}\sum_{k\in\mathcal{C}_t}(\tau(k)+1)^{-p}\Delta_k(\boldsymbol{\theta}_{t-\tau(k)}) = \sum_{s=1}^{t}\frac{(\tau+1)^{-p}}{C}\sum_{k\in\mathcal{K}_{t,s}}\Delta_k(\boldsymbol{\theta}_s) \\
&= \sum_{s=1}^{t}(\tau+1)^{-p}\frac{C_{t,s}}{C}(\boldsymbol{d}_s^\star + \frac{1}{C_{t,s}}\sum_{k\in\mathcal{K}_{t,s}}\Delta_k(\boldsymbol{\theta}_s) - \boldsymbol{d}_s^\star) = \sum_{s=1}^{t}(\tau+1)^{-p}\frac{C_{t,s}}{C}(\boldsymbol{d}_s^\star + \boldsymbol{\zeta}_{t,s}),
\end{aligned} \tag{5}$$

where $(\tau+1)^{-p}$ is a down-scaling factor commonly used in AsyncFL to mitigate the impact of staleness $\tau = t - s$ on model updates [41, 43, 56], and $\boldsymbol{d}_s^\star$ has the same definition as of in the

synchronous setting above. Besides, since the server only receives a subset of $C_{t,s}$ individual local updates out of the sampled K clients at iteration s, there is an extra sampling error denoted as $\boldsymbol{\zeta}_{t,s}$. We can define a weight matrix similar to $\boldsymbol{M}$:

$$\boldsymbol{W}_{[t,s]} = \begin{cases} (t-s+1)^{-p} C_{t,s}/C & \text{if } t >= s \\ 0 & \text{otherwise.} \end{cases}$$

Then, one can easily derive that $\boldsymbol{R} = \boldsymbol{D}^\star \boldsymbol{W}^\top + \boldsymbol{E}$ where the t-th column of error matrix $\boldsymbol{E}$ is defined as $\sum_{s=1}^{t} (\tau+1)^{-p} \frac{C_{t,s}}{C} \boldsymbol{\zeta}_{t,s}$. Substituting $\boldsymbol{R}$ back into (3), we get

$$\begin{aligned} \boldsymbol{\theta}_{T+1}^{\text{async}} =& \boldsymbol{\theta}_1 - \eta \boldsymbol{D}^\star \boldsymbol{W}^\top \boldsymbol{M}^\top \mathbf{1} - \eta \boldsymbol{E} \boldsymbol{M}^\top \mathbf{1} \\ =& \boldsymbol{\theta}_1 - \eta \boldsymbol{D}^\star \left[\boldsymbol{M}^\top + \underbrace{(\boldsymbol{M}\boldsymbol{W} - \boldsymbol{M})^\top}_{\text{implicit momentum bias}} \right] \mathbf{1} - \underbrace{\eta \boldsymbol{E} \boldsymbol{M}^\top \mathbf{1}}_{\text{async. sampling bias}} . \end{aligned} \tag{6}$$

Comparing the update rules Equations (4) and (6), there are two additional terms in asynchronous setting. One is the implicit momentum bias: the algorithm implicitly assigns biased weights $\boldsymbol{M}\boldsymbol{W}$ (which is different from normal momentum weight $\boldsymbol{M}$) to historical gradients, losing the benefits of momentum acceleration. We provide a visualization of $\boldsymbol{M}$ and $\boldsymbol{M}\boldsymbol{W}$ in Figure 2. While the normal momentum assigns the largest weight to the most recent gradients, asynchronous momentum tends to weigh more towards stale gradients, as they arrive more frequently. The second additional term in Equation (6) is the asynchronous sampling bias: the server sampled K clients from s-th iteration but can only received $C_{t,s}$ from the cohort at iteration t.

Previous works [39] also observed that giving additional lower weights (e.g. set $p > 0$) to historical gradients can help convergence. This phenomenon can be intuitively explained by the definition of implicit momentum bias, which becomes smaller when $\boldsymbol{W}$ approaches to the identity matrix. However, this approach cannot entirely solve the problem. It is nearly impossible to set $\boldsymbol{W} = \boldsymbol{I}$ in realistic settings, as the current gradients may only arrive in future iterations.

Theorem 3.6. *For SyncFL and AsyncFL with momentum, by choosing $\eta = \mathcal{O}(\frac{1}{\sqrt{T}})$,*

$$\mathbb{E}\|\frac{1}{T}(\boldsymbol{\theta}_{T+1}^\star - \boldsymbol{\theta}_{T+1}^{\text{sync}})\|_2^2 \leq \frac{1}{2}\eta^2 T G^2 = \mathcal{O}(G^2), \tag{7}$$

$$\mathbb{E}\|\frac{1}{T}(\boldsymbol{\theta}_{T+1}^\star - \boldsymbol{\theta}_{T+1}^{\text{async}})\|_2^2 \leq \eta^2 (2TS^2 + TG^2 + 2\frac{\rho^2}{C}) = \mathcal{O}(S^2 + G^2). \tag{8}$$

We defer the proof to Appendix B.1. Both SyncFL and AsyncFL have the same sampling bias in the order of $\mathcal{O}(G^2)$ on average, but AsyncFL introduces an extra $\mathcal{O}(S^2)$ term due to the implicit momentum bias. Note that $\boldsymbol{\theta}_{T+1}^\star$ can be different for SyncFL and AsyncFL due to different parameter update trajectory, and we focus on comparing to the $\boldsymbol{\theta}_{T+1}^\star$ within each algorithm.

3.2 Proposed Method: Momentum Approximation

From Equation (5), note that, in asynchronous setting, the received (pseudo-) gradient $\boldsymbol{r}_t$ is already a weighted average of historical gradients. Therefore, instead of naively applying momentum on top of it, can we simply adjust the weights to imitate the momentum updates? Following this idea, we propose a new update rule for AsyncFL:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \boldsymbol{R} \boldsymbol{a}_t, \tag{9}$$

where $\boldsymbol{a}_t \in \mathbb{R}^T$ is an arbitrary vector weighting the aggregated model updates. Accordingly, we have

$$\begin{aligned} \boldsymbol{\theta}_{T+1}^{\text{MA}} &= \boldsymbol{\theta}_1 - \eta \boldsymbol{R} \boldsymbol{A}^\top \mathbf{1} = \boldsymbol{\theta}_1 - \eta \boldsymbol{D}^\star \boldsymbol{W}^\top \boldsymbol{A}^\top \mathbf{1} - \eta \boldsymbol{E} \boldsymbol{A}^\top \mathbf{1} \\ &= \boldsymbol{\theta}_1 - \eta \boldsymbol{D}^\star [\boldsymbol{M}^\top + (\boldsymbol{A}\boldsymbol{W} - \boldsymbol{M})^\top] \mathbf{1} - \eta \boldsymbol{E} \boldsymbol{A}^\top \mathbf{1}, \end{aligned} \tag{10}$$

where $\boldsymbol{A}^\top = [\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_T]$. One can choose a matrix $\boldsymbol{A}$ such that $\boldsymbol{A}\boldsymbol{W} \approx \boldsymbol{M}$. As a result, the implicit momentum bias is largely removed. The resulting algorithm approximates the synchronous momentum method without explicitly adjusting momentum. For this reason, we name the proposed method as *momentum approximation (MA)*.

Implementation. The practical implementation of the proposed algorithm (outlined in Algorithm 1) is very straightforward. Thanks to the lower-triangular nature of both matrices $\boldsymbol{W}$ and $\boldsymbol{M}$, we can approximate the momentum matrix $\boldsymbol{M}$ row-by-row, i.e., in an online fashion. At iteration t, the desired weights for past gradients are given as the t-th row of $\boldsymbol{M}$ and known beforehand. We seek to optimize the following objective to find the best $\boldsymbol{a}_t = \boldsymbol{a}_{\text{opt}}$ to be used in Equation (9):

$$\min_{\boldsymbol{a}\in\mathbb{R}^T} \|\boldsymbol{a}^\top \boldsymbol{W} - \boldsymbol{M}_{[t,:]}\|_2^2, \text{ subject to } \boldsymbol{a}_{[s]} = 0, \forall s > t. \tag{11}$$

In each vector $\boldsymbol{a}_t$, only the first t elements are non-zero such that matrix $\boldsymbol{A}$ is enforced to be an lower-triangular matrix. This is because the server cannot use gradients from future iterations. Solving Equation (11) requires knowing $\boldsymbol{W}_{[:t,:t]}$ (the first t rows and columns of $\boldsymbol{W}$) which can be obtained by having each received client k to upload a one-hot encoding $\boldsymbol{e}_s \in \{0,1\}^T$ of their model version s.[2] More concretely, suppose at iteration t, the received updates at server are from a subset of C clients and their model version are denoted as $\{t-\tau(k)\}_{k\in\mathcal{C}_t}$. Then, the matrix $\boldsymbol{W}$ is initialized with all 0 and updated online as $\boldsymbol{W}_{[t,:]} = \frac{1}{C}\sum_{k\in\mathcal{C}_t} \boldsymbol{e}_{t-\tau(k)}^\top$. Sending the extra $\boldsymbol{e}_{t-\tau(k)}$ adds a negligible communication cost as $\boldsymbol{e}_{t-\tau(k)}$ has a payload size of T bits and $T \ll d$ for common FL tasks.

Theorem 3.7. *Under the condition that $\boldsymbol{W}$ is full rank and $\mathbb{E}\|\boldsymbol{A}\|_F^2 = \mathcal{O}(CT^2)$, for AsyncFL with momentum approximation (MA), by choosing $\eta = \mathcal{O}(\frac{1}{\sqrt{T}})$,*

$$\mathbb{E}\|\frac{1}{T}(\boldsymbol{\theta}^\star_{T+1} - \boldsymbol{\theta}^{\text{MA}}_{T+1})\|_2^2 \leq \eta^2(TG^2 + 2\frac{\rho^2}{TC}\mathbb{E}\|\boldsymbol{A}\|_F^2) = \mathcal{O}(G^2). \tag{12}$$

We defer the proof and discuss the more general case when $\boldsymbol{W}$ is not full rank to Appendix B.1. Under the given condition, AsyncFL with momentum approximation achieves the same error as SyncFL and drops the implicit momentum bias term in Equation (8). We show that the condition of $\mathbb{E}\|\boldsymbol{A}\|_F^2 = \mathcal{O}(CT^2)$ holds empirically in Appendix B.2.

Light-weight momentum approximation. The full approximation above requires a server-side storage cost of $\mathcal{O}(Td)$ as all past received gradients $\boldsymbol{R}$ needs to be saved to disk. This is usually not a concern as disk storage is cheap. In addition, T in FL is typically in the order of thousands and the model size d is small to meet on-device resource constraints [58, 59].

In the case of both T and d are high and the disk storage cost becomes a concern, we propose a light-weight approximation which has no extra storage cost on the server. The light-weight update rule is the same as Equation (9) except that $\boldsymbol{a}_t$ is replaced by $\tilde{\boldsymbol{a}}_t$ defined recursively as below:

$$\tilde{\boldsymbol{a}}_t = u_t\boldsymbol{e}_t + v_t\tilde{\boldsymbol{a}}_{t-1}, \tag{13}$$

where $u_t, v_t \in \mathbb{R}$ are to be optimized. With the recursive definition of $\tilde{\boldsymbol{a}}_t$, we rewrite Equation (9) as:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\boldsymbol{R}\tilde{\boldsymbol{a}}_t = \boldsymbol{\theta}_t - \eta(u_t\boldsymbol{R}\boldsymbol{e}_t + v_t\boldsymbol{R}\tilde{\boldsymbol{a}}_{t-1}) = \boldsymbol{\theta}_t - \eta(u_t\boldsymbol{r}_t + v_t\boldsymbol{R}\tilde{\boldsymbol{a}}_{t-1}), \tag{14}$$

which can be simplified to the following update rule similar to Equation (1):

$$\tilde{\boldsymbol{m}}_t = \boldsymbol{R}\tilde{\boldsymbol{a}}_t = u_t\boldsymbol{r}_t + v_t\tilde{\boldsymbol{m}}_{t-1},\ \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\tilde{\boldsymbol{m}}_t. \tag{15}$$

The difference to Equation (1) is that there are T pairs of real numbers (u_t, v_t) in light-weight MA instead of a single $\beta \in [0,1]$. Since Equation (15) depends on $\tilde{\boldsymbol{m}}$ and not on $\boldsymbol{R}$, light-weight approximation saves the extra $\mathcal{O}(Td)$ storage cost and has the same space complexity as the standard momentum updates by maintaining a single buffer $\tilde{\boldsymbol{m}}_t$.

To find the best $(u_t, v_t) = (u_{\text{opt}}, v_{\text{opt}})$ in iteration t, we substitute (13) back into (11):

$$\min_{u,v\in\mathbb{R}} \|(u\boldsymbol{e}_t + v\tilde{\boldsymbol{a}}_{t-1})^\top \boldsymbol{W} - \boldsymbol{M}_{[t,:]}\|_2^2. \tag{16}$$

Differentially private momentum approximation. Both the model updates Δ_k and the model version one-hot encoding $\boldsymbol{e}_{t-\tau(k)}$ are sensitive information as they reveal the client's local data and their timing of participating FL. We can use DP mechanisms to protect both information.

[2]We need to send the one-hot encoding instead of the integer $t-\tau(k)$ to server as one-hot encoding is compatible with secure aggregation and DP to update $\boldsymbol{W}$ and raw integer is not.

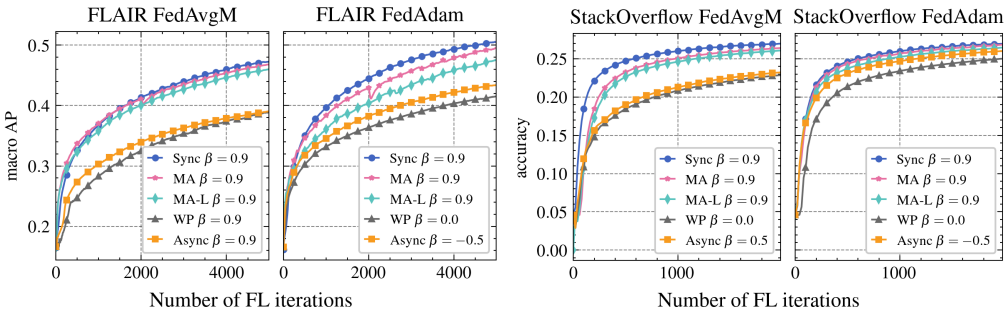

Figure 3: Comparison between MA, light-weight MA (MA-L) and baseline approaches.

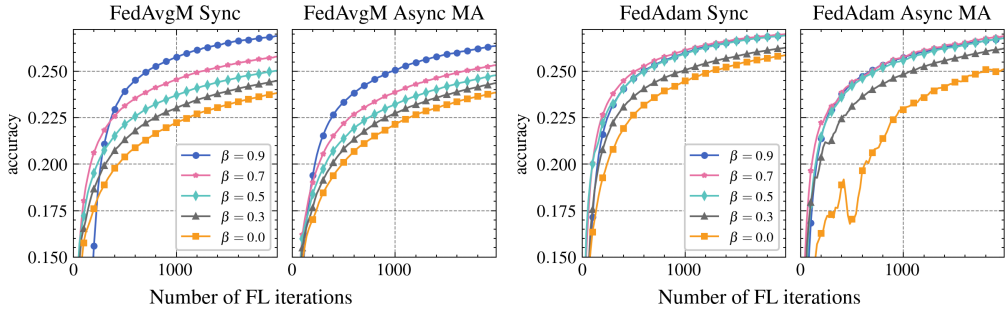

Figure 4: Impact of β on SyncFL and AsyncFL with MA on the StackOverflow dataset.

Let γ be a scaling factor and $\boldsymbol{y}_k = \Delta_k \oplus \gamma \boldsymbol{e}_{t-\tau(k)} \in \mathbb{R}^{d+T}$ be the payload that client k intends to send to the server, where $\oplus$ denotes vector concatenation. We constrain the L^2 sensitivity of $\boldsymbol{y}_k$ as $\bar{\boldsymbol{y}}_k = \text{clip}(\Delta_k, S_\Delta) \oplus \gamma \boldsymbol{e}_{t-\tau(k)}$, such that $\|\bar{\boldsymbol{y}}_k\|_2 \leq \sqrt{S_\Delta^2 + \gamma^2} = S$. Applying Gaussian Mechanism as $\sum_k \bar{\boldsymbol{y}}_k + \mathcal{N}(0, \sigma^2 S^2 \boldsymbol{I})$ satisfies (ϵ, δ)-DP as described in Section 2. By choosing $\gamma = \frac{\sigma}{\sqrt{\xi^2 - \sigma^2}} S_\Delta$, the Gaussian noise added to the un-scaled $\sum_k \boldsymbol{e}_{t-\tau(k)}$ is $\mathcal{N}(0, (\frac{\sigma}{\gamma} S)^2 \boldsymbol{I}))$ with standard deviation:

$$\frac{\sigma}{\gamma} S = \frac{\sigma}{\gamma} \sqrt{S_\Delta^2 + \gamma^2} = \sqrt{\frac{\sigma^2(\xi^2 - \sigma^2)}{\sigma^2} + \sigma^2} = \xi. \tag{17}$$

In practice, we tune $\xi > \sigma$ to balance the utility on $\sum_k \Delta_k$ and $\sum_k \boldsymbol{e}_{t-\tau(k)}$. As momentum approximation is a post-processing [17] on the private aggregates:

$$\boldsymbol{r}_t = \frac{1}{C}[\sum_{k \in \mathcal{C}_t} \text{clip}(\Delta_k, S_\Delta) + \mathcal{N}(0, \sigma^2 S^2 \boldsymbol{I})], \; \boldsymbol{W}_{[t,:]} = \frac{1}{C\gamma}[\sum_{k \in \mathcal{C}_t} \gamma \boldsymbol{e}_{t-\tau(k)}^\top + \mathcal{N}(0, \sigma^2 S^2 \boldsymbol{I})], \tag{18}$$

the MA update rules in Equations (9) and (15) also satisfies the same (ϵ, δ)-DP guarantee.

Implicit momentum bias in the preconditioner. For adaptive optimizers such as Adam [26] and RMSProp, the preconditioner $\boldsymbol{H}_t$ is the square root of exponentially decaying average of the gradients' second moments: $\text{diag}(\boldsymbol{H}_t)^2 \leftarrow \beta' \text{diag}(\boldsymbol{H}_t)^2 + (1 - \beta') \boldsymbol{r}_t^2$. The stale updates in $\boldsymbol{r}_t$ bias the estimation of second moments similar to the implicit momentum bias term in the first moments. Nonetheless, preconditioner is known to be robust to delayed gradients [18, 30]. In Figure 1, we show that β' impacts the performance in AsyncFL similarly to in SyncFL. We leave the study of implicit bias from staleness in the second moments to future work.

4 Experiments

In this section, we describe the empirical evaluation of momentum approximation (MA) with FedBuff. We denote the light-weight MA in Equation (16) as MA-light or MA-L. We focus on two server-side momentum-based optimizers: FedAvgM [21] and FedAdam [45].

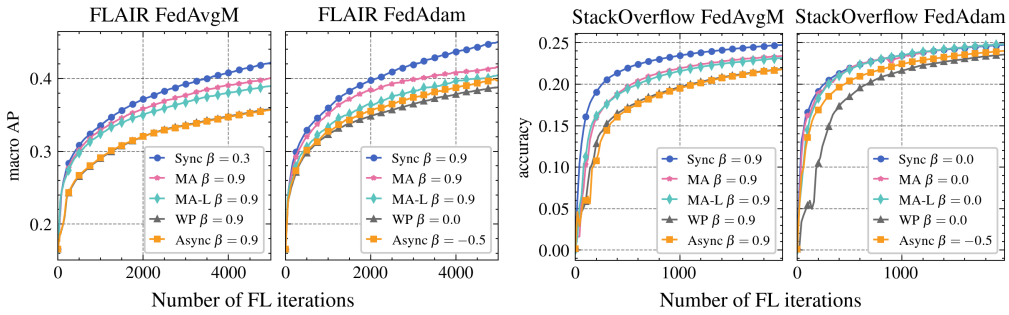

Figure 5: Comparison between MA, light-weight MA (MA-L) and baseline approaches with DP.

Datasets and ML Tasks. We conduct experiments on FLAIR [46], a large-scale annotated image dataset for multi-label classification, and StackOverflow [1], a commonly used language modeling FL benchmark dataset. Both datasets have natural client partition which captures the non-IID characteristics in real world FL setting, and we believe they better represent the production FL datasets compared to other commonly used datasets (e.g. CIFAR10) with artificially simulated client partition from a given distribution (e.g. Dirichlet [21]).

For the FLAIR dataset, the task is to predict the set of coarse-grained labeled objects in a given image. We use macro averaged precision (macro AP) as the evaluation metric. For the StackOverflow dataset, the task is next word prediction and we use top prediction accuracy as the evaluation metric following prior work [45]. The details of hyerparameter choices are described in Appendix A.1

4.1 Baselines

AsyncFL with tuned momentum parameter. We consider FedBuff as the baseline AsyncFL approach. As suggested in [39, 41], β needs to be tuned carefully in AsyncFL and sometimes negative β performs better. We tune β from the range $(-1, 1)$.

Weight prediction (WP). WP is proposed to speed up AsyncSGD [27] and in particular, to address the implicit momentum issue [19] in traditional distributed training setting. We modify WP to be compatible with AsyncFL as detailed in Appendix A.2.

4.2 Results

Figure 3 summarizes the convergence comparison between our proposed MA and the baseline approaches. For both FedAvgM and FedAdam on both datasets, MA and MA-light significantly outperforms the baseline approaches of best tuned β. We do not find WP worked well in the FL setting and acknowledge that more thoughtful integration is required to adopt techniques from AsyncSGD literature to AsyncFL, which is beyond the scope of this work.

Impact of β. Figure 4 illustrates how β impacts SyncFL and AsyncFL with MA. The correlation pattern between β and the performance remains the same between SyncFL and AsyncFL with MA, i.e., larger β leads to better performance in this task. This demonstrates that the AsyncFL with MA can reuse the tuned β in SyncFL experiments instead of searching in a wider range from scratch, which saves the costs from expensive hyperparameter tuning in FL.

Impact of cohort size. Smaller cohort can negatively impact the convergence of MA as the variance of $\boldsymbol{r}_t$ in Equation (5) increases. We evaluate the impact of cohort size C on the StackOverflow dataset by varying C from 50 to 400. We compare the performance between SyncFL and FedBuff with MA on the same C. Left of Table 1 shows the impact of C on MA. As C increases from 50 to 400, the gap between AsyncFL with MA and SyncFL becomes smaller, which validates our hypothesis that smaller cohort size has more negative impacts on MA than SyncFL.

DP results. Figure 5 illustrates the convergence results with DP. The pattern of the performance comparison is similar to that of the non-private case where both MA and MA-light outperform the AsyncFL baselines. We notice that in FedAdam baseline, negative β values are optimal on both datasets, indicating that the baseline requires more hyper-parameter tuning in a wider range of β.

Table 1: (Left) relative accuracy gap (%) between SyncFL and AsyncFL with MA for different cohort sizes C on the StackOverflow dataset. (Right) Relative speed up ($\times$) of MA compared to FedBuff baseline. FLR denotes FLAIR and SO denotes StackOverflow.

	FedAvgM		FedAam	
C	MA	MA-light	MA	MA-light
50	5.17	7.68	7.26	11.94
100	3.17	4.25	2.47	4.51
200	2.18	3.44	0.52	1.87
400	1.39	2.8	0.46	0.4

	FedAvgM		FedAam	
Setup	MA	MA-light	MA	MA-light
FLR	3.56	3.01	2.20	1.66
FLR w. DP	2.57	2.09	1.61	1.15
SO	3.96	3.30	1.62	1.30
SO w. DP	2.06	1.80	1.50	1.55

Speed up of MA. We finally evaluate the speed up of MA. Following [41], we record the iterations that MA needed to achieve the best metric from FedBuff baseline, and have this number divided by the iterations the baseline took to get the relative speed up. Right of Table 1 summarizes the results. On both FLAIR and StackOverflow, MA speeds up FedAvgM more than FedAdam, and we suspect the reason is that the preconditioner in the FedAdam baseline is less affected, thereby mitigating the impact of staleness. Thus the improvement from MA is less significant. DP also impacts the speed up negatively as the estimation of $\boldsymbol{W}$ becomes noisy and the noise scale on $\boldsymbol{r}_t$ increases.

5 Related Work

Asynchronous distributed SGD. The negative impact of gradient staleness has been studied in the traditional AsyncSGD setting. A line of research focused on reducing the impact of staleness or the discrepancy between worker's model and the central model. [63] first proposed to down-scale the stale gradients based on their staleness τ. [2] argued that τ failed to accurately reflect the discrepancy and proposed to schedule down-scale based on the similarity between worker's model and the central model. [64] used a Taylor expansion and Hessian approximation to compensate for the staleness. [19, 27] proposed parameter prediction of the future central model to reduce discrepancy, simply by following the optimization oracle for more iterations. The impact of staleness is exacerbated by momentum, as analyzed in [39], where a small or even negative momentum parameter is preferred to adverse effects of asynchrony, which motivated an adaptive momentum training schedule [62].

Asynchronous federated learning. Existing AsyncFL works draw inspirations from AsyncSGD to handle stragglers and heterogeneous latency. [41, 43, 56] down-scaled the local model updates based on staleness τ before the central aggregation. Nonetheless, as we showed, stale updates after down-scaling still affect training performance of momentum in practice.

On the other hand, FL differs from the traditional distributed SGD setting by having massive number of clients, higher communication overhead and more complicated system for secure aggregation and DP. Hence, many AsyncSGD algorithms do not directly fit in AsyncFL. Gradient compensation [64] applied complicated operations on individual client updates, which is less obvious how to implement with secure aggregation. [8, 19, 27] required broadcasting model parameters and momentum, leading to doubled communication cost. Further research is needed to study how to integrate the other promising AsyncSGD algorithms efficiently in AsyncFL.

Apart from aforementioned AsyncFL algorithms on weight aggregation, there are many orthogonal tactics on improving AsyncFL. For example, gradient compression techniques [28, 33] improved communication efficiency; and model splitting [7, 13, 54] had each client responsible for training a certain part of the whole model.

Momentum-based federated optimizers. [21] first proposed to extend FedAvg with server-side momentum (FedAvgM) to accelerate convergence. [45] improved the server-side optimization further by using adaptive optimizers with momentum. [24, 25, 42, 57] proposed to perform momentum updates locally on the clients to alleviate local drift problem. [47] introduced multistage FedGM which interpolates between FedAvg and FedAvgM with a hyperparameter scheduler, and provides a general momentum computation for FL to better control the momentum acceleration. Our approach is compatible to any momentum-based optimizers and orthogonal to these works as they were not proposed to resolve the server-side implicit momentum bias in AsyncFL.

6 Conclusion

We demonstrate how stale model updates incur an implicit bias in AsyncFL, which diminishes the acceleration from momentum-based optimizers. To address this issue, we propose momentum approximation which optimizes a least square problem online to find the optimal weighted average of historical model updates that approximates the desired momentum updates. Momentum approximation is easy to integrate in production FL systems with a minor storage and communication cost. We empirically evaluate momentum approximation in both non-private and private settings on real-world benchmark FL datasets, and demonstrated that it outperforms the existing AsyncFL algorithms.

References

[1] The Tensorflow Federated Authors. Tensorflow federated stack overflow dataset. `https://www.tensorflow.org/federated/api_docs/python/tff/simulation/datasets/stackoverflow/load_data`, 2019.

[2] Saar Barkai, Ido Hakimi, and Assaf Schuster. Gap aware mitigation of gradient staleness. *arXiv preprint arXiv:1909.10802*, 2019.

[3] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, Brendan McMahan, et al. Towards federated learning at scale: System design. *Proceedings of machine learning and systems*, 1:374–388, 2019.

[4] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[5] Han Cai, Ji Lin, Yujun Lin, Zhijian Liu, Haotian Tang, Hanrui Wang, Ligeng Zhu, and Song Han. Enable deep learning on mobile devices: Methods, systems, and applications. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 27(3):1–50, 2022.

[6] Zheng Chai, Yujing Chen, Ali Anwar, Liang Zhao, Yue Cheng, and Huzefa Rangwala. Fedat: A high-performance and communication-efficient federated learning system with asynchronous tiers. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16, 2021.

[7] Yang Chen, Xiaoyan Sun, and Yaochu Jin. Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation. *IEEE transactions on neural networks and learning systems*, 31(10):4229–4238, 2019.

[8] Yujing Chen, Yue Ning, Martin Slawski, and Huzefa Rangwala. Asynchronous online federated learning for edge devices with non-iid data. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 15–24. IEEE, 2020.

[9] Christopher A Choquette-Choo, Arun Ganesh, Ryan McKenna, H Brendan McMahan, Keith Rush, Abhradeep Guha Thakurta, and Zheng Xu. (amplified) banded matrix factorization: A unified approach to private training. *Advances in Neural Information Processing Systems*, 2023.

[10] Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022.

[11] Sergey Denisov, H Brendan McMahan, John Rush, Adam Smith, and Abhradeep Guha Thakurta. Improved differential privacy for sgd via optimal private linear operators on adaptive streams. *Advances in Neural Information Processing Systems*, 35:5910–5924, 2022.

[12] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[13] Chen Dun, Mirian Hipolito, Chris Jermaine, Dimitrios Dimitriadis, and Anastasios Kyrillidis. Efficient and light-weight federated learning via asynchronous distributed dropout. In *International Conference on Artificial Intelligence and Statistics*, pages 6630–6660. PMLR, 2023.

[14] Sanghamitra Dutta, Jianyu Wang, and Gauri Joshi. Slow and stale gradients can win the race. *IEEE Journal on Selected Areas in Information Theory*, 2(3):1012–1024, 2021.

[15] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Advances in Cryptology-EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28-June 1, 2006. Proceedings 25*, pages 486–503. Springer, 2006.

[16] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer, 2006.

[17] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[18] Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning*, pages 1842–1850. PMLR, 2018.

[19] Ido Hakimi, Saar Barkai, Moshe Gabel, and Assaf Schuster. Taming momentum in a distributed asynchronous environment. *arXiv preprint arXiv:1907.11612*, 2019.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[22] Dzmitry Huba, John Nguyen, Kshitiz Malik, Ruiyu Zhu, Mike Rabbat, Ashkan Yousefpour, Carole-Jean Wu, Hongyuan Zhan, Pavel Ustinov, Harish Srinivas, et al. Papaya: Practical, private, and scalable federated learning. *Proceedings of Machine Learning and Systems*, 4:814–832, 2022.

[23] Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling. In *International Conference on Machine Learning*, pages 5213–5225. PMLR, 2021.

[24] Sai Praneeth Karimireddy, Martin Jaggi, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2020.

[25] Prashant Khanduri, Pranay Sharma, Haibo Yang, Mingyi Hong, Jia Liu, Ketan Rajawat, and Pramod Varshney. Stem: A stochastic two-sided momentum algorithm achieving near-optimal sample and communication complexities for federated learning. *Advances in Neural Information Processing Systems*, 34:6050–6061, 2021.

[26] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[27] Atli Kosson, Vitaliy Chiley, Abhinav Venigalla, Joel Hestness, and Urs Koster. Pipelined backpropagation at scale: training large models without batches. *Proceedings of Machine Learning and Systems*, 3:479–501, 2021.

[28] Ming Li, Yiwei Chen, Yiqin Wang, and Yu Pan. Efficient asynchronous vertical federated learning via gradient prediction and double-end sparse compression. In *2020 16th international conference on control, automation, robotics and vision (ICARCV)*, pages 291–296. IEEE, 2020.

[29] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[30] Tian Li, Manzil Zaheer, Ken Liu, Sashank J Reddi, Hugh Brendan McMahan, and Virginia Smith. Differentially private adaptive optimization with delayed preconditioners. In *International Conference on Learning Representations*, 2023.

[31] Ji Lin, Wei-Ming Chen, Yujun Lin, Chuang Gan, Song Han, et al. Mcunet: Tiny deep learning on iot devices. *Advances in Neural Information Processing Systems*, 33:11711–11722, 2020.

[32] Ji Lin, Ligeng Zhu, Wei-Ming Chen, Wei-Chen Wang, Chuang Gan, and Song Han. On-device training under 256kb memory. *Advances in Neural Information Processing Systems*, 35:22941–22954, 2022.

[33] Xiaofeng Lu, Yuying Liao, Pietro Lio, and Pan Hui. Privacy-preserving asynchronous federated learning mechanism for edge network computing. *IEEE Access*, 8:48970–48981, 2020.

[34] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[35] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

[36] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 691–706. IEEE, 2019.

[37] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*, pages 263–275. IEEE, 2017.

[38] Ilya Mironov, Kunal Talwar, and Li Zhang. R\'enyi differential privacy of the sampled gaussian mechanism. *arXiv preprint arXiv:1908.10530*, 2019.

[39] Ioannis Mitliagkas, Ce Zhang, Stefan Hadjis, and Christopher Ré. Asynchrony begets momentum, with an application to deep learning. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 997–1004. IEEE, 2016.

[40] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 739–753. IEEE, 2019.

[41] John Nguyen, Kshitiz Malik, Hongyuan Zhan, Ashkan Yousefpour, Mike Rabbat, Mani Malek, and Dzmitry Huba. Federated learning with buffered asynchronous aggregation. In *International Conference on Artificial Intelligence and Statistics*, pages 3581–3607. PMLR, 2022.

[42] Emre Ozfatura, Kerem Ozfatura, and Deniz Gündüz. Fedadc: Accelerated federated learning with drift control. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 467–472. IEEE, 2021.

[43] Jungwuk Park, Dong-Jun Han, Minseok Choi, and Jaekyun Moon. Sageflow: Robust federated learning against both stragglers and adversaries. *Advances in neural information processing systems*, 34:840–851, 2021.

[44] Matthias Paulik, Matt Seigel, Henry Mason, Dominic Telaar, Joris Kluivers, Rogier van Dalen, Chi Wai Lau, Luke Carlson, Filip Granqvist, Chris Vandevelde, et al. Federated evaluation and tuning for on-device personalization: System design & applications. *arXiv preprint arXiv:2102.08503*, 2021.

[45] Sashank J Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2020.

[46] Congzheng Song, Filip Granqvist, and Kunal Talwar. Flair: Federated learning annotated image repository. *Advances in Neural Information Processing Systems*, 35:37792–37805, 2022.

[47] Jianhui Sun, Xidong Wu, Heng Huang, and Aidong Zhang. On the role of server momentum in federated learning. *arXiv preprint arXiv:2312.12670*, 2023.

[48] Kunal Talwar, Shan Wang, Audra McMillan, Vojta Jina, Vitaly Feldman, Bailey Basile, Aine Cahill, Yi Sheng Chan, Mike Chatzidakis, Junye Chen, et al. Samplable anonymous aggregation for private federated data analysis. *arXiv preprint arXiv:2307.15017*, 2023.

[49] Marten van Dijk, Nhuong V Nguyen, Toan N Nguyen, Lam M Nguyen, Quoc Tran-Dinh, and Phuong Ha Nguyen. Asynchronous federated learning with reduced number of rounds and with differential privacy from less aggregated gaussian noise. *arXiv preprint arXiv:2007.09208*, 2020.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[51] Ewen Wang, Boyi Chen, Mosharaf Chowdhury, Ajay Kannan, and Franco Liang. Flint: A platform for federated learning integration. *Proceedings of Machine Learning and Systems*, 5, 2023.

[52] Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

[53] Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat. Slowmo: Improving communication-efficient distributed sgd with slow momentum. *arXiv preprint arXiv:1910.00643*, 2019.

[54] Qizhao Wang, Qing Li, Kai Wang, Hong Wang, and Peng Zeng. Efficient federated learning for fault diagnosis in industrial cloud-edge computing. *Computing*, 103(10):2319–2337, 2021.

[55] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K Leung, Christian Makaya, Ting He, and Kevin Chan. Adaptive federated learning in resource constrained edge computing systems. *IEEE journal on selected areas in communications*, 37(6):1205–1221, 2019.

[56] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*, 2019.

[57] Jing Xu, Sen Wang, Liwei Wang, and Andrew Chi-Chih Yao. Fedcm: Federated learning with client-level momentum. *arXiv preprint arXiv:2106.10874*, 2021.

[58] Mingbin Xu, Congzheng Song, Ye Tian, Neha Agrawal, Filip Granqvist, Rogier van Dalen, Xiao Zhang, Arturo Argueta, Shiyi Han, Yaqiao Deng, et al. Training large-vocabulary neural language models by private federated learning for resource-constrained devices. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[59] Zheng Xu, Yanxiang Zhang, Galen Andrew, Christopher A Choquette-Choo, Peter Kairouz, H Brendan McMahan, Jesse Rosenstock, and Yuanbo Zhang. Federated learning of gboard language models with differential privacy. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, 2023.

[60] Haibo Yang, Minghong Fang, and Jia Liu. Achieving linear speedup with partial worker participation in non-iid federated learning. *International Conference on Learning Representations*, 2021.

[61] Feilong Zhang, Xianming Liu, Shiyi Lin, Gang Wu, Xiong Zhou, Junjun Jiang, and Xiangyang Ji. No one idles: Efficient heterogeneous federated learning with parallel edge and server computation. In *International Conference on Machine Learning*, pages 41399–41413. PMLR, 2023.

[62] Jian Zhang and Ioannis Mitliagkas. Yellowfin and the art of momentum tuning. *arXiv preprint arXiv:1706.03471*, 2017.

[63] Wei Zhang, Suyog Gupta, Xiangru Lian, and Ji Liu. Staleness-aware async-sgd for distributed deep learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2350–2356, 2016.

[64] Shuxin Zheng, Qi Meng, Taifeng Wang, Wei Chen, Nenghai Yu, Zhi-Ming Ma, and Tie-Yan Liu. Asynchronous stochastic gradient descent with delay compensation. In *International Conference on Machine Learning*, pages 4120–4129. PMLR, 2017.

[65] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in neural information processing systems*, 32, 2019.

Algorithm 1 FedBuff with Momentum Approximation

Inputs: client sampling rate q, cohort size C, server optimizer SERVEROPT, server learning rate η, number of FL iterations T, client local learning rate η_l, number, number of client local SGD steps Q
for $t = 1, \dots T$ **do**
  $\mathcal{K}_t \leftarrow$ sampled K clients with sampling rate q
  Run CLIENT($\boldsymbol{\theta}_t, t$) for $k \in \mathcal{K}_t$ asynchronously
  if receives $\Delta_k(\boldsymbol{\theta}_{t-\tau(k)})$ and $\boldsymbol{e}_{t-\tau(k)}$ from k **then**
    $\boldsymbol{r}_t \leftarrow \boldsymbol{r}_t + \frac{1}{C}\Delta_k(\boldsymbol{\theta}_{t-\tau(k)})$
    $\boldsymbol{W}_{[t,:]} \leftarrow \boldsymbol{W}_{[t,:]} + \frac{1}{C}\boldsymbol{e}_{t-\tau(k)}^\top$
  if server received C local model updates **then**
    if light-weight momentum approximation **then**
      $u_t, v_t \leftarrow$ solve Equation (16)
      $\tilde{\boldsymbol{m}}_t \leftarrow v_t \tilde{\boldsymbol{m}}_{t-1} + u_t \boldsymbol{r}_t$
    else
      $\boldsymbol{R}_{[:,t]} \leftarrow \boldsymbol{r}_t$ update the pseudo-gradient history
      $\boldsymbol{a}_t \leftarrow$ solve Equation (11)
      $\tilde{\boldsymbol{m}}_t \leftarrow \boldsymbol{R}\boldsymbol{a}_t$
    $\boldsymbol{H}_t \leftarrow$ update based on SERVEROPT
    $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \eta \boldsymbol{H}_t^{-1}\tilde{\boldsymbol{m}}_t$
    $\boldsymbol{r}_{t+1}, \boldsymbol{W}_{[t+1,:]} \leftarrow \boldsymbol{0}$

function CLIENT($\boldsymbol{\theta}, t$)
  $\boldsymbol{\theta}' \leftarrow$ run Q SGD steps with η_l on local data
  $\boldsymbol{e}_t \leftarrow$ one-hot encoding of t
  Upload $\Delta = \boldsymbol{\theta} - \boldsymbol{\theta}'$ and $\boldsymbol{e}_t$ to server

A Additional Experiments Details

A.1 Experimental Setup

All experiments were run on a machine with 4 Nvidia A100 GPUs with 40GB VRAM. Each FLAIR experiment took 12 hours to finish on average and each StackOverflow experiment took 6 hours.

Client delay distribution. Following [41], we adopt half-Normal distribution to model the client delay distribution. We demonstrate the impact of different distributions on MA in Appendix A.3.

Staleness scaling and bounding. We tune the power in the down-scaling factor p between 0.5 to 2.0 to control strength of the scaling. We further set a maximum staleness bound $\tau_{\max}$ (default to 20) and drop Δ_k if $\tau(k) > \tau_{\max}$.

Hyperparameters. For the FLAIR dataset, we use a ResNet-18 model [20] following the setup in [46]. We train the model for 5,000 iterations with local learning rate set to 0.1, local epochs set to 2, and local batch size set to 16. For the StackOverflow dataset, we use a 3-layer Transformer model [50] following the setup in [52]. We train the model for 2,000 iterations with local learning rate set to 0.3 and local epochs set to 1, and local batch size set to 16. The cohort size C is set to 200 for both dataset. For FedAvgM, we search the learning rate η between (0.1, 1.0). For FedAdam, we set the β' for the second moment to 0.99 and the adaptivity parameter to 0.01, and search the server learning rate η between (0.01, 0.1).

For DP experiments, we set the (ϵ, δ) privacy budget to $(2.0, 10^{-7})$-DP with a simulated population size of 10^7 and cohort size of 5,000 following prior work [35]. We set the L^2 clipping bound S_Δ to 0.1 for FLAIR and 0.2 for StackOverflow. We use amplification by subsampling with Rényi DP to calibrate the Gaussian noise scale σ [37, 38]. Though we focus on independent Gaussian mechanism in each iteration, our approach is also compatible with DP-FTRL mechanisms with correlated noise between iterations [9, 23]. For momentum approximation where $\boldsymbol{W}$ needs to be estimate privately, we set ξ in Equation (17) such that $S = 1.1 S_\Delta$, i.e. we pay 10% extra noise on Δ to learn $\boldsymbol{W}$ privately with the same budget.

Algorithm 2 FedBuff with Weight Prediction

Inputs: client Poisson sampling rate q, cohort size C, server optimizer SERVEROPT, server learning rate η, number of FL iterations T, client local learning rate η_l, number, number of client local SGD steps Q, α EMA decay parameter for historical model updates

while $t < T$ **do**
  $\mathcal{C}_t \leftarrow$ sampled clients with Poisson sampling rate q
  Run CLIENT$(\theta_t, t, \eta \boldsymbol{H}_t^{-1} \boldsymbol{x}_h)$ for $k \in \mathcal{C}_t$ asynchronously
  if receives $\Delta_k(\theta_{t-\tau(k)})$ from k **then**
    $\boldsymbol{x}_t \leftarrow \boldsymbol{x}_t + \frac{1}{C}\Delta_k(\theta_{t-\tau(k)})$
  if received C results in the buffer **then**
    $\boldsymbol{m}_t, \boldsymbol{H}_t \leftarrow$ update based on SERVEROPT
    $\theta_{t+1} \leftarrow \theta_t - \eta \boldsymbol{H}_t^{-1} \boldsymbol{m}_t$
    $\boldsymbol{x}_h \leftarrow \alpha \boldsymbol{x}_h + (1-\alpha)\boldsymbol{x}_t$
    $\boldsymbol{x}_{t+1} \leftarrow \mathbf{0}$
    $t \leftarrow t+1$

function CLIENT$(\theta, t, \eta \boldsymbol{H}_t^{-1} \boldsymbol{x}_h)$
  $\tau \leftarrow t' - t$ gets the current staleness
  $\hat{\theta}_{t+\tau} \leftarrow \theta - \tau \eta \boldsymbol{H}_t^{-1} \boldsymbol{x}_h$
  $\theta' \leftarrow$ run Q SGD steps with η_l on $\hat{\theta}_{t+\tau}$
  Upload $\Delta = \hat{\theta}_{t+\tau} - \theta'$ to server

Table 2: (Left) Accuracy (%) with momentum approximation (MA) for different staleness bound $\tau_{\max}$ on the StackOverflow dataset. (Right) Relative least square error in Equation (11) for different client delay distribution.

	FedAvgM		FedAam	
$\tau_{\max}$	MA	MA-light	MA	MA-light
20	26.36	26.11	26.79	26.45
30	26.26	26.05	26.54	26.24
40	26.15	25.95	26.25	25.87
50	25.97	25.90	26.00	25.46

Client Delay Distribution	MA	MA-light
Half-Normal	2.58%	33.07%
Uniform	8.35%	36.78%
Exponential	2.41%	33.89%

For all experiments, we apply exponential moving average (EMA) on central model parameters θ with decay rate of 0.99 [10], and report the metrics evaluated on the EMA model parameters.

A.2 Weight Prediction Baseline

WP is proposed to speed up asynchronous SGD [27] and in particular, to address the implicit momentum issue [19]. In Algorithm 2, we modify WP to be compatible with adaptive optimizer in FedBuff as another baseline for evaluating momentum approximation. To predict the future model, the server sends both θ_t and the historical model updates $\boldsymbol{x}_h$ to devices. For a sampled client with staleness τ, the client tries to first predict the future model $\hat{\theta}_{t+\tau} \approx \theta_{t+\tau}$, by running τ steps of SERVEROPT$(\theta, \boldsymbol{x}_h, \eta)$. We consider $\boldsymbol{x}_h$ to be the exponential decay averaging of $\boldsymbol{X}_{:t}$ for variance reduction. For adaptive SERVEROPT such as Adam, we send $\boldsymbol{H}_t^{-1}\boldsymbol{x}_h$ to devices for WP. Client then runs the local SGD steps on $\hat{\theta}_{t+\tau}$ and returns the model update to the server. Note that this method will also double the communication as the server needs to send extra historical model updates for WP.

A.3 Additional Results

Impact of staleness bound $\tau_{\max}$. We empirically study the impact of $\tau_{\max}$ on momentum approximation by varying it from 20 to 50. Left of Table 2 summarizes the results on the StackOverflow dataset, where larger $\tau_{\max}$ leads to lower accuracy. Another observation is that the drop in performance of FedAdam is greater than that of FedAvgM which could be from the impact of staleness on the estimation of preconditioner in FedAdam.

Impact of client delay distribution. We evaluate the impact of different client delay distribution on momentum approximation objective in Equation (11). We choose Half-Normal, Uniform and Exponential distribution following [41]. We measure the relative least square error as $\|\boldsymbol{AW} - \boldsymbol{M}\|_F^2/\|\boldsymbol{M}\|_F^2$ using the setup in Appendix A.1 and report the results in the right of Table 2. The relative error for light-weight approximation is much higher as expected and is more than 30% for all three distributions. The approximation error is worst for Uniform distribution, while it is in the similar range for Half-Normal and Exponential distribution. Uniform client delay distribution is unrealistic in production FL system and thus our method is robust to different sensible client delay distributions.

B Additional Details of Momentum Approximation

Notation. Let $\mathcal{H}_i$ be the history of sampled clients $\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_i$ up to iteration i. Let $\boldsymbol{\zeta}_s = \boldsymbol{d}_s - \boldsymbol{d}_s^\star$ be the sampling error for the subset $\mathcal{K}_s$ sampled at iteration s. Let $\boldsymbol{d}_{t,s} = \frac{1}{C_{t,s}} \sum_{k \in \mathcal{K}_{t,s}} \Delta_k(\boldsymbol{\theta}_s)$ be the averaged update of the set of clients $\mathcal{K}_{t,s} \subseteq \mathcal{K}_s$ whose updates arrived at iteration t and denote $C_{t,s} = |\mathcal{K}_{t,s}|$. Let $\boldsymbol{\zeta}_{t,s} = \boldsymbol{d}_{t,s} - \boldsymbol{d}_s^\star$ be the sampling error for the subset $\mathcal{K}_{t,s}$. Let $\mathcal{K}_{:i,:i}$ be the subsets $\mathcal{K}_{t,s}$ for all $1 \le s \le t \le i$ and $C_{:i,:i}$ be their cardinalities.

We first note an upper bound on the sampling error $\boldsymbol{\zeta}_s$. From Assumption 3.2,

$$\begin{aligned}
\mathbb{E}_{\mathcal{K}_s}[\|\boldsymbol{\zeta}_s\|_2^2] &= \mathbb{E}_{\mathcal{K}_s}[\|\frac{1}{K}\sum_{k \in \mathcal{K}_s} \Delta_k(\boldsymbol{\theta}_s) - \boldsymbol{d}_s^\star\|_2^2] \\
&\le \mathbb{E}_{\mathcal{K}_s}[\frac{1}{K^2} K \sum_{k \in \mathcal{K}_s} \|\Delta_k(\boldsymbol{\theta}_s) - \boldsymbol{d}_s^\star\|_2^2] \\
&= \frac{1}{K}\sum_{j=1}^{K} \mathbb{E}_{k \sim [m]}[\|\Delta_k(\boldsymbol{\theta}_s) - \boldsymbol{d}_s^\star\|_2^2] \le G^2
\end{aligned} \tag{19}$$

B.1 Proof of Results

We state two useful lemmas for proving Theorem 3.6 and 3.7.

Lemma B.1. *For any* $1 \le s \le t \le i$ *and* $1 \le s' \le t' \le i$*,*

$$\mathbb{E}_{\mathcal{K}_{t,s}, \mathcal{K}_{t',s'} | C_{t,s}, C_{t',s'}, \mathcal{H}_i}[\boldsymbol{\zeta}_{t,s}^\top \boldsymbol{\zeta}_{t',s'}] \le \boldsymbol{\zeta}_s^\top \boldsymbol{\zeta}_{s'} + \frac{\rho^2}{C_{t,s}} \mathbb{1}[(t,s) = (t',s')]. \tag{20}$$

Proof. Let $\boldsymbol{\zeta}_{t,s,k} = \Delta_k(\boldsymbol{\theta}_s) - \boldsymbol{d}_s^\star$ for $k \in \mathcal{K}_{t,s}$, and given Assumption 3.4, we have

$$\mathbb{E}_{k \sim \mathcal{K}_{t,s} | \mathcal{H}_i}[\boldsymbol{\zeta}_{t,s,k}] = \mathbb{E}_{k \sim \mathcal{K}_s | \mathcal{H}_i}[\Delta_k(\boldsymbol{\theta}_s)] - \boldsymbol{d}_s^\star = \boldsymbol{d}_s - \boldsymbol{d}_s^\star = \boldsymbol{\zeta}_s. \tag{21}$$

Then for the case when $(t,s) \ne (t',s')$,

$$\begin{aligned}
&\mathbb{E}_{\mathcal{K}_{t,s}, \mathcal{K}_{t',s'} | C_{t,s}, C_{t',s'}, \mathcal{H}_i}[\boldsymbol{\zeta}_{t,s}^\top \boldsymbol{\zeta}_{t',s'}] \\
&= \mathbb{E}_{\mathcal{K}_{t,s}, \mathcal{K}_{t',s'} | C_{t,s}, C_{t',s'}, \mathcal{H}_i}[\frac{1}{C_{t,s} C_{t',s'}} \sum_{k \in \mathcal{K}_{t,s}} \sum_{k' \in \mathcal{K}_{t',s'}} \boldsymbol{\zeta}_{t,s,k}^\top \boldsymbol{\zeta}_{t',s',k'}] \\
&= \frac{1}{C_{t,s} C_{t',s'}} \sum_{j=1}^{C_{t,s}} \sum_{j'=1}^{C_{t',s'}} \mathbb{E}_{k \sim \mathcal{K}_{t,s} | \mathcal{H}_i}[\boldsymbol{\zeta}_{t,s,k}^\top] \mathbb{E}_{k' \sim \mathcal{K}_{t',s'} | \mathcal{H}_i}[\boldsymbol{\zeta}_{t',s',k'}] \\
&= \frac{1}{C_{t,s} C_{t',s'}} \sum_{j=1}^{C_{t,s}} \sum_{j'=1}^{C_{t',s'}} \boldsymbol{\zeta}_s^\top \boldsymbol{\zeta}_{s'} = \boldsymbol{\zeta}_s^\top \boldsymbol{\zeta}_{s'}.
\end{aligned} \tag{22}$$

For the case when $(t,s)=(t',s')$,

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{K}_{t,s},\mathcal{K}_{t',s'}|C_{t,s},C_{t',s'},\mathcal{H}_i}[\boldsymbol{\zeta}_{t,s}^\top \boldsymbol{\zeta}_{t',s'}] \\
&= \mathbb{E}_{\mathcal{K}_{t,s}|C_{t,s},\mathcal{H}_i}[\boldsymbol{\zeta}_{t,s}^\top \boldsymbol{\zeta}_{t,s}] = \mathbb{E}_{\mathcal{K}_{t,s}|C_{t,s},\mathcal{H}_i}[\frac{1}{C_{t,s}^2}\sum_{k,k'\in\mathcal{K}_{t,s}} \boldsymbol{\zeta}_{t,s,k}^\top \boldsymbol{\zeta}_{t,s,k'}] \\
&= \mathbb{E}_{\mathcal{K}_{t,s}|C_{t,s},\mathcal{H}_i}[\frac{1}{C_{t,s}^2}\sum_{k,k'\in\mathcal{K}_{t,s},k\neq k'} \boldsymbol{\zeta}_{t,s,k}^\top \boldsymbol{\zeta}_{t,s,k'} + \frac{1}{C_{t,s}^2}\sum_{k\in\mathcal{K}_{t,s}} \|\boldsymbol{\zeta}_{t,s,k}\|_2^2] \\
&= \frac{C_{t,s}^2 - C_{t,s}}{C_{t,s}^2}\mathbb{E}_{k\sim\mathcal{K}_{t,s}|\mathcal{H}_i}[\boldsymbol{\zeta}_{t,s,k}^\top]\mathbb{E}_{k'\sim\mathcal{K}_{t,s}|\mathcal{H}_i}[\boldsymbol{\zeta}_{t,s,k'}] + \frac{1}{C_{t,s}}\mathbb{E}_{k\sim\mathcal{K}_{t,s}|\mathcal{H}_i}\|\boldsymbol{\zeta}_{t,s,k}\|_2^2 \\
&\leq \frac{C_{t,s}^2 - C_{t,s}}{C_{t,s}^2}\boldsymbol{\zeta}_s^\top \boldsymbol{\zeta}_s + \frac{1}{C_{t,s}}(\|\boldsymbol{\zeta}_s\|_2^2 + \rho^2) = \boldsymbol{\zeta}_s^\top \boldsymbol{\zeta}_s + \frac{\rho^2}{C_{t,s}}.
\end{aligned} \tag{23}
$$

□

Lemma B.2. *For any coefficient $x_{i,t}\in\mathbb{R}$,*

$$
\mathbb{E}_{\mathcal{K}_{:i,:i}|\mathcal{H}_i}\|\sum_{t=1}^{i} x_{i,t}\sum_{s=1}^{i}\boldsymbol{W}_{[t,s]}\boldsymbol{\zeta}_{t,s}\|_2^2 \leq \mathbb{E}_{C_{:i,:i}|\mathcal{H}_i}[\|\sum_{t=1}^{i} x_{i,t}\sum_{s=1}^{i}\boldsymbol{W}_{[t,s]}\boldsymbol{\zeta}_s\|_2^2 + \frac{\rho^2}{C}\sum_{t=1}^{i}x_{i,t}^2]. \tag{24}
$$

Proof.

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{K}_{:i,:i}|\mathcal{H}_i}\|\sum_{t=1}^{i} x_{i,t}\sum_{s=1}^{i}\boldsymbol{W}_{[t,s]}\boldsymbol{\zeta}_{t,s}\|_2^2 \\
&= \mathbb{E}_{\mathcal{K}_{:i,:i}|\mathcal{H}_i}[\sum_{\substack{t=1\\s=1}}^{i}\sum_{\substack{t'=1\\s'=1}}^{i} x_{i,t}x_{i,t'}\boldsymbol{W}_{[t,s]}\boldsymbol{W}_{[t',s']}\boldsymbol{\zeta}_{t,s}^\top\boldsymbol{\zeta}_{t',s'}] \\
&= \mathbb{E}_{C_{:i,:i}|\mathcal{H}_i}[\sum_{\substack{t=1\\s=1}}^{i}\sum_{\substack{t'=1\\s'=1}}^{i} x_{i,t}x_{i,t'}\boldsymbol{W}_{[t,s]}\boldsymbol{W}_{[t',s']}\mathbb{E}_{\mathcal{K}_{t,s},\mathcal{K}_{t',s'}|C_{t,s},C_{t',s'},\mathcal{H}_i}[\boldsymbol{\zeta}_{t,s}^\top\boldsymbol{\zeta}_{t',s'}]] \\
&\overset{\text{Lemma B.1}}{\leq} \mathbb{E}_{C_{:i,:i}|\mathcal{H}_i}[\sum_{\substack{t=1\\s=1}}^{i}\sum_{\substack{t'=1\\s'=1}}^{i} x_{i,t}x_{i,t'}\boldsymbol{W}_{[t,s]}\boldsymbol{W}_{[t',s']}\boldsymbol{\zeta}_s^\top\boldsymbol{\zeta}_{s'} + \sum_{\substack{t=1\\s=1}}^{i} x_{i,t}^2\boldsymbol{W}_{[t,s]}^2\rho^2\frac{1}{C_{t,s}}] \\
&= \mathbb{E}_{C_{:i,:i}|\mathcal{H}_i}[\|\sum_{t=1}^{i} x_{i,t}\sum_{s=1}^{i}\boldsymbol{W}_{[t,s]}\boldsymbol{\zeta}_s\|_2^2 + \frac{\rho^2}{C}\sum_{t=1}^{i}x_{i,t}^2\sum_{s=1}^{i}\boldsymbol{W}_{[t,s]}] \\
&\leq \mathbb{E}_{C_{:i,:i}|\mathcal{H}_i}[\|\sum_{t=1}^{i} x_{i,t}\sum_{s=1}^{i}\boldsymbol{W}_{[t,s]}\boldsymbol{\zeta}_s\|_2^2 + \frac{\rho^2}{C}\sum_{t=1}^{i}x_{i,t}^2].
\end{aligned} \tag{25}
$$

The last inequality comes from the fact that $\sum_{s=1}^{t}\boldsymbol{W}_{[t,s]}\leq 1$. □

Theorem B.3. *For SyncFL and AsyncFL with momentum,*

$$
\mathbb{E}\|\frac{1}{T}(\boldsymbol{\theta}_{T+1}^\star - \boldsymbol{\theta}_{T+1}^{\text{sync}})\|_2^2 \leq \frac{1}{2}\eta^2 TG^2, \tag{26}
$$

$$
\mathbb{E}\|\frac{1}{T}(\boldsymbol{\theta}_{T+1}^\star - \boldsymbol{\theta}_{T+1}^{\text{async}})\|_2^2 \leq \eta^2(2TS^2 + TG^2 + 2\frac{\rho^2}{C}). \tag{27}
$$

Proof. For SyncFL with momentum,

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{\theta}^\star_{T+1} - \boldsymbol{\theta}^{\text{sync}}_{T+1}\|_2^2 &= \eta^2\mathbb{E}\|(\boldsymbol{D}-\boldsymbol{D}^\star)\boldsymbol{M}^\top\mathbf{1}\|_2^2 \\
&\le \eta^2 T\mathbb{E}\|(\boldsymbol{D}-\boldsymbol{D}^\star)\boldsymbol{M}^\top\|_F^2 \\
&= \eta^2 T\sum_{i=1}^{T}\mathbb{E}\|\boldsymbol{M}_{[i,:]}(\boldsymbol{D}-\boldsymbol{D}^\star)\|_2^2 \\
&= \eta^2 T\sum_{i=1}^{T}\mathbb{E}\|\sum_{t=1}^{i}\boldsymbol{M}_{[i,t]}\boldsymbol{\zeta}_t\|_2^2 \\
&\le \eta^2 T\sum_{i=1}^{T} i\sum_{t=1}^{i}\boldsymbol{M}^2_{[i,t]}\mathbb{E}\|\boldsymbol{\zeta}_t\|_2^2 \\
&\le \eta^2 T\sum_{i=1}^{T} iG^2 \le \frac{1}{2}\eta^2T^3G^2.
\end{aligned} \tag{28}$$

By telescoping the constant, we get:

$$\mathbb{E}\|\frac{1}{T}(\boldsymbol{\theta}^\star_{T+1} - \boldsymbol{\theta}^{\text{sync}}_{T+1})\|_2^2 \le \frac{1}{2}\eta^2TG^2. \tag{29}$$

For AsyncFL with momentum,

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{\theta}^\star_{T+1} - \boldsymbol{\theta}^{\text{async}}_{T+1}\|_2^2 &= \eta^2\mathbb{E}\|\boldsymbol{D}^\star(\boldsymbol{M}\boldsymbol{W}-\boldsymbol{M})^\top\mathbf{1} + \boldsymbol{E}\boldsymbol{M}^\top\mathbf{1}\|_2^2 \\
&\le 2\eta^2(\mathbb{E}\|\boldsymbol{D}^\star(\boldsymbol{M}\boldsymbol{W}-\boldsymbol{M})^\top\mathbf{1}\|_2^2 + \mathbb{E}\|\boldsymbol{E}\boldsymbol{M}^\top\mathbf{1}\|_2^2) \\
&\le 2\eta^2T(\mathbb{E}\|\boldsymbol{D}^\star(\boldsymbol{M}\boldsymbol{W}-\boldsymbol{M})^\top\|_F^2 + \mathbb{E}\|\boldsymbol{E}\boldsymbol{M}^\top\|_F^2)
\end{aligned} \tag{30}$$

Let $\check{\boldsymbol{M}} = \boldsymbol{M}\boldsymbol{W}$ where $\check{\boldsymbol{M}}_{[i,:]} = \boldsymbol{M}_{[i,:]}\boldsymbol{W}$ and $\sum_{t=1}^{i}\check{\boldsymbol{M}}_{[i,t]} \le \sum_{t=1}^{i}\boldsymbol{M}_{[i,t]}$ given $\sum_{s=1}^{t}\boldsymbol{W}_{[t,s]} \le 1$. Then $\|\check{\boldsymbol{M}}_{[i,:]}\|_2 \le \|\check{\boldsymbol{M}}_{[i,:]}\|_1 = \|\boldsymbol{M}_{[i,:]}\|_1 \le 1$. We first bound the implicit momentum bias term.

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{D}^\star(\boldsymbol{M}\boldsymbol{W}-\boldsymbol{M})^\top\|_F^2 &= \sum_{i=1}^{T}\mathbb{E}\|(\check{\boldsymbol{M}}_{[i,:]} - \boldsymbol{M}_{[i,:]})\boldsymbol{D}^{\star\top}\|_2^2 \\
&= \sum_{i=1}^{T}\mathbb{E}\|\sum_{t=1}^{i}(\check{\boldsymbol{M}}_{[i,t]} - \boldsymbol{M}_{[i,t]})\boldsymbol{d}^\star_t\|_2^2 \\
&\le \sum_{i=1}^{T}\mathbb{E}[i\sum_{t=1}^{i}(\check{\boldsymbol{M}}_{[i,t]} - \boldsymbol{M}_{[i,t]})^2\|\boldsymbol{d}^\star_t\|_2^2] \\
&\le \sum_{i=1}^{T} iS^2\mathbb{E}[\|\check{\boldsymbol{M}}_{[i,:]} - \boldsymbol{M}_{[i,:]}\|_2^2] \\
&\le S^2\sum_{i=1}^{T} i\mathbb{E}[\|\check{\boldsymbol{M}}_{[i,:]}\|_2^2 + \|\boldsymbol{M}_{[i,:]}\|_2^2 - 2\check{\boldsymbol{M}}_{[i,:]}\boldsymbol{M}^\top_{[i,:]}] \\
&\le S^2\sum_{i=1}^{T} 2i \le S^2T^2.
\end{aligned} \tag{31}$$

We then bound the asynchronous sampling bias term:

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{E}\boldsymbol{M}^\top\|_F^2 &= \sum_{i=1}^{T}\mathbb{E}\|\boldsymbol{M}_{[i,:]}\boldsymbol{E}^\top\|_2^2 \\
&= \sum_{i=1}^{T}\mathbb{E}_{\mathcal{H}_i}\mathbb{E}_{\mathcal{K}_{:i,:i}|\mathcal{H}_i}\|\sum_{t=1}^{i}\boldsymbol{M}_{[i,t]}\sum_{s=1}^{i}\boldsymbol{W}_{[t,s]}\boldsymbol{\zeta}_{t,s}\|_2^2 \\
&\overset{\text{Lemma B.2}}{\leq} \sum_{i=1}^{T}\mathbb{E}_{\mathcal{H}_i}\mathbb{E}_{C_{1:i,1:i}|\mathcal{H}_i}[\|\sum_{t=1}^{i}\sum_{s=1}^{i}\boldsymbol{M}_{[i,t]}\boldsymbol{W}_{[t,s]}\boldsymbol{\zeta}_s\|_2^2 + \frac{\rho^2}{C}\sum_{t=1}^{i}\boldsymbol{M}_{[i,t]}^2] \\
&= \sum_{i=1}^{T}\mathbb{E}[\|\sum_{s=1}^{i}\boldsymbol{\zeta}_s\sum_{t=1}^{i}\boldsymbol{M}_{[i,t]}\boldsymbol{W}_{[t,s]}\|_2^2 + \frac{\rho^2}{C}\|\boldsymbol{M}_{[i:]}\|_2^2] \\
&\leq \sum_{i=1}^{T}\mathbb{E}[\|\sum_{s=1}^{i}\check{\boldsymbol{M}}_{[i,s]}\boldsymbol{\zeta}_s\|_2^2 + \frac{\rho^2}{C}] \\
&\leq \sum_{i=1}^{T}\mathbb{E}[iG^2\|\check{\boldsymbol{M}}_{[i,:]}\|_2^2] + T\frac{\rho^2}{C} \\
&\leq \sum_{i=1}^{T} iG^2 + T\frac{\rho^2}{C} \leq \frac{1}{2}G^2T^2 + T\frac{\rho^2}{C}.
\end{aligned} \tag{32}$$

By substituting Equation (31) and (32) in Equation (30) and telescoping the constant, we get:

$$\mathbb{E}\|\frac{1}{T}(\boldsymbol{\theta}_{T+1}^\star - \boldsymbol{\theta}_{T+1}^{\text{async}})\|_2^2 \leq \eta^2(2TS^2 + TG^2 + 2\frac{\rho^2}{C}). \tag{33}$$

□

We next present a generalized version of Theorem 3.7 for $\boldsymbol{W}$ with any rank. We note that at iteration i, the first i elements of solution $\boldsymbol{a}_i[:i] = \boldsymbol{M}_{[i,:i]}\boldsymbol{W}_{[:i,:i]}^+$ where $\boldsymbol{W}_{[:i,:i]}^+$ is the Moore–Penrose inverse of $\boldsymbol{W}_{[:i,:i]}$, and the rest elements in $\boldsymbol{a}_i[:i]$ are zeros. Let r_i denotes the rank of $\boldsymbol{W}_{[:i,:i]}$, and let $[\boldsymbol{U}_{r_i}, \boldsymbol{U}_{-r_i}]\boldsymbol{\Sigma}_i[\boldsymbol{V}_{r_i}, \boldsymbol{V}_{-r_i}]^\top$ be the singular value decomposition of $\boldsymbol{W}_{[:i,:i]}$, where $\boldsymbol{U}_{r_i}, \boldsymbol{V}_{r_i}$ are the first r_i left and right singular vectors ordered by the singular values.

We define

$$\alpha_i = \frac{\boldsymbol{M}_{[i,:i]}\boldsymbol{V}_{r_i}\boldsymbol{V}_{r_i}^\top\boldsymbol{M}_{[i,:i]}^\top}{\|\boldsymbol{M}_{[i,:]}\|_2^2} \in [0,1],$$

i.e. the normalized magnitude of projection of $\boldsymbol{M}_{[i,:]}$ onto the columns space of $\boldsymbol{W}_{[:i,:i]}$. Then we can decompose $\|\boldsymbol{M}_{[i,:]}\|_2^2$ as:

$$\begin{aligned}
\|\boldsymbol{M}_{[i,:]}\|_2^2 &= \boldsymbol{M}_{[i,:i]}\boldsymbol{V}_{r_i}\boldsymbol{V}_{r_i}^\top\boldsymbol{M}_{[i,:i]}^\top + \boldsymbol{M}_{[i,:i]}\boldsymbol{V}_{-r_i}\boldsymbol{V}_{-r_i}^\top\boldsymbol{M}_{[i,:i]}^\top \\
&= \alpha_i\|\boldsymbol{M}_{[i,:]}\|_2^2 + \boldsymbol{M}_{[i,:i]}\boldsymbol{V}_{-r_i}\boldsymbol{V}_{-r_i}^\top\boldsymbol{M}_{[i,:i]}^\top
\end{aligned}$$

Thus,

$$\boldsymbol{M}_{[i,:i]}\boldsymbol{V}_{-r_i}\boldsymbol{V}_{-r_i}^\top\boldsymbol{M}_{[i,:i]}^\top = (1-\alpha_i)\|\boldsymbol{M}_{[i,:i]}\|_2^2 \leq 1-\alpha_i. \tag{34}$$

Theorem B.4. *Let $A(T) = \sum_{i=1}^{T} i\mathbb{E}[1-\alpha_i]$. For AsyncFL with momentum approximation (MA),*

$$\mathbb{E}\|\frac{1}{T}(\boldsymbol{\theta}_{T+1}^\star - \boldsymbol{\theta}_{T+1}^{\text{MA}})\|_2^2 \leq \eta^2(2S^2\frac{A(T)}{T} + G^2T + 2\frac{\rho^2}{TC}\mathbb{E}\|\boldsymbol{A}\|_F^2). \tag{35}$$

Proof. Similar to Equation (30), we have:

$$\mathbb{E}\|\boldsymbol{\theta}_{T+1}^\star - \boldsymbol{\theta}_{T+1}^{\text{MA}}\|_2^2 \leq 2\eta^2T(\mathbb{E}\|\boldsymbol{D}^\star(\boldsymbol{A}\boldsymbol{W} - \boldsymbol{M})^\top\|_F^2 + \mathbb{E}\|\boldsymbol{E}\boldsymbol{A}^\top\|_F^2) \tag{36}$$

Denote $\tilde{\boldsymbol{M}} = \boldsymbol{A}\boldsymbol{W}$ and $\tilde{\boldsymbol{M}}_{[i,:i]} = \boldsymbol{a}_i[:i]^\top \boldsymbol{W}_{[:i,:i]} = \boldsymbol{M}_{[i,:i]}\boldsymbol{W}^+_{[:i,:i]}\boldsymbol{W}_{[:i,:i]}$. We first bound the implicit momentum bias term:

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{D}^\star(\boldsymbol{A}\boldsymbol{W}-\boldsymbol{M})^\top\|_F^2 &= \sum_{i=1}^{T}\mathbb{E}\|(\tilde{\boldsymbol{M}}_{[i,:]}-\boldsymbol{M}_{[i,:]})\boldsymbol{D}^{\star\top}\|_2^2 \\
&\le \sum_{i=1}^{T} iS^2\mathbb{E}\|\tilde{\boldsymbol{M}}_{[i,:i]}-\boldsymbol{M}_{[i,:i]}\|_2^2 \\
&= \sum_{i=1}^{T} iS^2\mathbb{E}\|\boldsymbol{M}_{[i,:i]}(\boldsymbol{W}^+_{[:i,:i]}\boldsymbol{W}_{[:i,:i]}-\boldsymbol{I})\|_2^2 \\
&= \sum_{i=1}^{T} iS^2\mathbb{E}\|\boldsymbol{M}_{[i,:i]}\boldsymbol{V}_{-r_i}\boldsymbol{V}_{-r_i}^\top\|_2^2 \\
&= \sum_{i=1}^{T} iS^2\mathbb{E}[\boldsymbol{M}_{[i,:i]}\boldsymbol{V}_{-r_i}\boldsymbol{V}_{-r_i}^\top\boldsymbol{M}_{[i,:i]}^\top] \\
&\le \sum_{i=1}^{T} iS^2\mathbb{E}[1-\alpha_i] = S^2A(T)
\end{aligned} \tag{37}$$

We then bound the asynchronous sampling bias term. Since $\boldsymbol{W}^+\boldsymbol{W}$ is a orthogonal projection operator, $\|\tilde{\boldsymbol{M}}_{[i,:]}\|_2^2 \le \|\boldsymbol{M}_{[i,:]}\|_2^2 \le 1$ and we have:

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{E}\boldsymbol{A}^\top\|_F^2 &= \sum_{i=1}^{T}\mathbb{E}\|\boldsymbol{A}_{[i,:]}\boldsymbol{E}^\top\|_2^2 \\
&= \sum_{i=1}^{T}\mathbb{E}_{\mathcal{H}_i}\mathbb{E}_{\mathcal{K}_{:i,:i}|\mathcal{H}_i}[\|\sum_{t=1}^{i}\boldsymbol{A}_{[i,t]}\sum_{s=1}^{i}\boldsymbol{W}_{[t,s]}\boldsymbol{\zeta}_{t,s}\|_2^2] \\
&\overset{\text{Lemma B.2}}{\le} \sum_{i=1}^{T}\mathbb{E}[\|\sum_{s=1}^{i}\tilde{\boldsymbol{M}}_{[i:]}\boldsymbol{\zeta}_s\|_2^2 + \frac{\rho^2}{C}\|\boldsymbol{a}_i\|_2^2] \\
&\le \sum_{i=1}^{T}\mathbb{E}[iG^2\|\tilde{\boldsymbol{M}}_{[i,:]}\|_2^2] + \frac{\rho^2}{C}\sum_{i=1}^{T}\mathbb{E}\|\boldsymbol{a}_i\|_2^2 \\
&\le \sum_{i=1}^{T} iG^2 + \frac{\rho^2}{C}\mathbb{E}\|\boldsymbol{A}\|_F^2 \le \frac{1}{2}T^2G^2 + \frac{\rho^2}{C}\mathbb{E}\|\boldsymbol{A}\|_F^2.
\end{aligned} \tag{38}$$

By substituting Equation (37) and (38) in Equation (36) and telescoping the constant, we get:

$$\mathbb{E}\|\frac{1}{T}(\boldsymbol{\theta}^\star_{T+1}-\boldsymbol{\theta}^{\text{MA}}_{T+1})\|_2^2 \le \eta^2(2S^2\frac{A(T)}{T} + G^2T + 2\frac{\rho^2}{TC}\mathbb{E}\|\boldsymbol{A}\|_F^2). \tag{39}$$

□

When $\boldsymbol{W}$ is full rank, $\alpha_i = 1$ for all iterations and $A(T) = 0$. When $\boldsymbol{W}$ is not full rank, α_i is more likely to be closer to 1 when r_i is high and leads to smaller $A(T)$. We discuss in Appendix B.2 that how can down-scaling factor increases r_i and reduces implicit momentum bias.

B.2 Discussion

Justification for Assumption 3.4. We argue that in the production FL systems, the order of sampled clients' updates arriving at server is random does not dependent on their data size. In common deployed FL systems [3, 22, 44, 51], the initiation of on-device training process is subject to

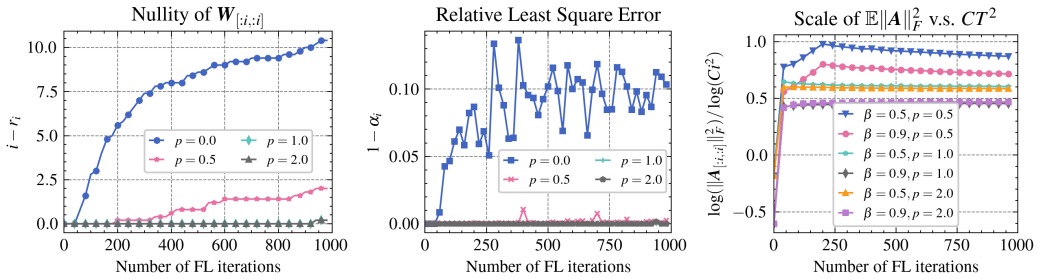

Figure 6: (Left) Nullity $(i - r_i)$ of $\boldsymbol{W}_{[:i,:i]}$ over iterations for different p. (Middle) $1 - \alpha_i$ over iterations for different p. (Right) Scale of $\mathbb{E}\|\boldsymbol{A}\|_F^2$ versus CT^2 in the log space over iterations for different p and β.

a set of conditions: connected to power and wireless network, idle, and scheduled by device OS. The timing of the event for all these conditions to be met is naturally nondeterministic rather than data dependent, e.g. a client might always charge their device at certain time of a day but OS scheduler might not always prioritize the FL training process and start training at exactly the time of charging.

Let X_k denote the random variable for client k's the starting time of training, and let Y_k be the random variable of on-device training and network latency for submitting model update to server. Then $X_k + Y_k$ is the time that k's updates arrived at server. When the variance of X_k is dominating $X_k + Y_k$, the arrival order of sampled clients is not data dependent and random. This is highly likely as the on-device training becomes extremely efficient with advances in hardware, e.g. training a modern neural network on an edge device takes only a few seconds [5, 31, 32]. To further enforce the arrival order to be random and less data dependent, we can also implement simple on-device logic such as enforcing the maximum amount of data to train on and injects a small random delay before on-device training [48].

Importance of the down-scaling factor. The down-scaling factor $(t - s + 1)^{-p}$ for the stale model updates plays an important role when solving Equation (11). We find that larger p leads to higher rank of $\boldsymbol{W}$ and smaller least square objective, and thus better momentum approximation as illustrated in Figure 6. For $p = 0.0$, the nullity $(i - r_i)$ increases over iterations while $\boldsymbol{W}_{[:i,:i]}$ is mostly full-rank when choosing a higher p. As a consequence, the relative least square error $1 - \alpha_i$ is nearly zero for higher p. However, we cannot set p arbitrarily high as this would over-penalize the stale gradients and impact the model convergence.

Scale of $\mathbb{E}\|\boldsymbol{A}\|_F^2$. We empirically evaluate the condition in Theorem 3.7 that $\mathbb{E}\|\boldsymbol{A}\|_F^2 = \mathcal{O}(CT^2)$, i.e. $\mathbb{E}\|\boldsymbol{A}\|_F^2$ should not grow faster than CT^2 when T increases. Right of Figure 6 shows the quantity $\log\|\boldsymbol{A}_{[:i,:i]}\|_F^2 / \log Ci^2$ over iterations, where the quantity is less than 1 for different choices of p and β, indicating that $\mathbb{E}\|\boldsymbol{A}\|_F^2$ grows slower than CT^2. We also note that higher p results in smaller $\mathbb{E}\|\boldsymbol{A}\|_F^2$ leading to smaller sampling error in Equation (12).

C Limitations

Our analysis in Theorem 3.6 and Theorem 3.7 did not make any assumptions about the distribution of $\boldsymbol{W}$. Though the analysis gives the results for an arbitrary $\boldsymbol{W}$ but we note that the distribution of $\boldsymbol{W}$ might have some special properties when using a particular client delay distribution. As discussed in Appendix B.2, we also empirically find the spectral properties of $\boldsymbol{W}$ are associated the choice of down-scaling factor and its exponent p. These special properties of $\boldsymbol{W}$, which we did not formalize, could potentially improve the theoretical results . We leave it to future work to explore the impact of the distribution and properties of $\boldsymbol{W}$ on momentum in AsyncFL. We also did not analyze the impact of stale gradients on the bias in the second moments in optimizers like FedAdam as we acknowledged in Section 3.2.

D Broader Impact

This work does not have negative societal or ethical impact. On the contrary, this work can potentially benefit the society in terms of stronger privacy protection. Our proposed method is compatible with secure aggregation and differential privacy, and can be easily integrated to existing asynchronous federated learning production systems. We believe that our method can improve the applicability of asynchronous private federated learning to more on-device ML products where the data is highly personal and sensitive, and thus provide meaningful privacy guarantee to the end users.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction are reflected in both theoretical and empirical results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: Limitations are discussed in Appendix C.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Assumptions are provided in Section 3 and proofs are provided in Appendix B.1.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Detailed description of hyperparameters and setup is provided in Appendix A.1. We plan to open source the code and data for reproducing the results in the paper in the near future.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We plan to open source the code and data for reproducing the results in the paper in the near future.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Detailed description of hyperparameters and setup is provided in Appendix A.1. We also separately analyzed the impact of some critical hyperparameters in the algorithms.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The statistical significance is not available momentarily as the experiments are expensive and time-consuming to run. We plan to include the error bars for all results in the next updated version.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The resources are described in Appendix A.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The authors have reviewed and the research conform with the the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The broader impacts of this paper is discussed in Appendix D.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This question is not relevant for this paper.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The authors have cited the original papers that produced the datasets.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.