# Beyond ADMM: A Unified Client-Variance-Reduced Adaptive Federated Learning Framework

**Shuai Wang**[1], **Yanqing Xu**[2], **Zhiguo Wang**[3], **Tsung-Hui Chang**[2] *, **Tony Q. S. Quek**[1], **Defeng Sun**[4]

[1] Information Systems Technology and Design, Singapore University of Technology and Design, 487372 Singapore
[2] School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen 518172, China
[3] College of Mathematics, Sichuan University, Chengdu, Sichuan 610064, China
[4] Department of Applied Mathematics, The Hong Kong Polytechnic University, Hong Kong
shuaiwang@link.cuhk.edu.cn, xuyanqing@cuhk.edu.cn, wangzhiguo@scu.edu.cn, changtsunghui@cuhk.edu.cn, tonyquek@sutd.edu.sg, defeng.sun@polyu.edu.hk

## Abstract

As a novel distributed learning paradigm, federated learning (FL) faces serious challenges in dealing with massive clients with heterogeneous data distribution and computation and communication resources. Various client-variance-reduction schemes and client sampling strategies have been respectively introduced to improve the robustness of FL. Among others, primal-dual algorithms such as the alternating direction of method multipliers (ADMM) have been found being resilient to data distribution and outperform most of the primal-only FL algorithms. However, the reason behind remains a mystery still. In this paper, we firstly reveal the fact that the federated ADMM is essentially a client-variance-reduced algorithm. While this explains the inherent robustness of federated ADMM, the vanilla version of it lacks the ability to be adaptive to the degree of client heterogeneity. Besides, the global model at the server under client sampling is biased which slows down the practical convergence. To go beyond ADMM, we propose a novel primal-dual FL algorithm, termed FedVRA, that allows one to adaptively control the variance-reduction level and biasness of the global model. In addition, FedVRA unifies several representative FL algorithms in the sense that they are either special instances of FedVRA or are close to it. Extensions of FedVRA to semi/un-supervised learning are also presented. Experiments based on (semi-)supervised image classification tasks demonstrate superiority of FedVRA over the existing schemes in learning scenarios with massive heterogeneous clients and client sampling.

## 1 Introduction

As a local-privacy-aware distributed paradigm, federated learning (FL) recently has drawn significant attention in the distributed machine learning (ML) community (Kŏnecný, McMahan, and Ramage 2015). In a typical FL setting, a central server coordinates $N$ distributed clients to jointly solve the following learning problem:

$$\min_{\mathbf{x}} \; f(\mathbf{x}) \triangleq \sum_{i=1}^{N} \omega_i f_i(\mathbf{x}), \tag{1}$$

where $f_i(\mathbf{x})$ is the (possibly non-convex) local cost function and $\omega_i$ is the weight coefficient associated with client $i$. Different from traditional distributed learning (Chang et al. 2020), FL faces numerous challenges such as limited communication resources, and data heterogeneity and system heterogeneity over massive clients (Yang et al. 2019; Kŏnecný et al. 2016). Many FL algorithms including the popular FedAvg (Li et al. 2020b; McMahan et al. 2017) adopt the partial client participation (PCP) strategy using client sampling and local stochastic gradient descent (local SGD) (Stich 2019) to overcome the network congestion problem and improve the communication efficiency. However, FedAvg is inefficient in dealing with data heterogeneity and system heterogeneity where the clients have non-i.i.d. local datasets and have varied computational capabilities, respectively (Li et al. 2020a). On one hand, the presence of non-i.i.d. data would cause the client drift issue which degrades the algorithm convergence performance and can even lead to model divergence (Li et al. 2020b; Karimireddy et al. 2020). On the other hand, heterogeneity in the computational speeds results in large variations in the number of local updates performed by each client (i.e., heterogeneous local updates (HLU)), which leads to solution bias and convergence slowdown (Wang et al. 2020). In addition, under PCP, many of the FL algorithms require unbiased client sampling, otherwise the global model at the server is biased and the algorithm cannot converge to a proper solution (Cho, Wang, and Joshi 2022).

To address the data heterogeneity issue, client-variance-reduction (CVR) schemes such as VRL-SGD (Liang et al. 2020) and SCAFFOLD (Karimireddy et al. 2020) have been proposed. They identified the inter-client variance [1] caused by non-i.i.d. data as the main reason for the client drift issue and attempted to alleviate it. Specifically, in the CVR schemes, control variates are introduced to perturb the local stochastic gradient (SG) so as to approximate the *global SG* (which assumes data samples are drawn from all client's data in the

*Corresponding author


[1] If the local data of all clients have an identical distribution, then all local stochastic gradients (SGs) have the same mean and it is equal to the global SG. However, when local data have different distributions, there is a gap between the local SGs and global SG, which is called inter-client variance. Such a notion is widely used in the FL literature.

centralized manner), thereby mitigating the client drift effect. For overcoming the system heterogeneity, FedNova (Wang et al. 2020) proposed the use of normalized averaging SGD to account for HLU. However, the aforementioned methods do not fully resolve both the data and system heterogeneity issues. For example, VRL-SGD and SCAFFOLD do not consider HLU to address the system heterogeneity, and FedProx and FedNova still suffer from convergence slowdown caused by non-i.i.d. data.

Interestingly, recent findings show that primal-dual FL methods based on the alternating direction method of multipliers (ADMM) (Boyd et al. 2010) (Hajinezhad et al. 2016) are inherently resilient to both data and system heterogeneity, see, e.g., FedPD (Zhang et al. 2021), FedADMM (Gong, Li, and Freris 2022) and FedDyn (Acar et al. 2021). However, their convergences rely on the constant and uniform client sampling, and the requirement of the clients to either solve the local subproblems globally or to a sufficient accuracy. Besides, it is not clear how the distributed ADMM algorithms are related to the existing CVR schemes. Table 1 summarizes the aforementioned FL algorithms with their client sampling strategies under PCP, convergence rate, and whether the clients adopt local SGD (LSGD) and HLU.

| Algorithm | PCP | LSGD | HLU | Convergence rate |
|---|---|---|---|---|
| FedAvg | UB | ✓ | ✗ | $\mathcal{O}\left(\frac{\sigma^2+G^2}{\sqrt{mR}}+\frac{G}{R^{2/3}}+\frac{B^2}{R}\right)$ |
| FedProx | UB | ✗ | ✓ | $\mathcal{O}(\frac{B^2}{R})$ |
| VRL-SGD | ✗ | ✓ | ✗ | $\mathcal{O}\left(\frac{\sigma^2}{S\sqrt{NR}}+\frac{N}{R}\right)$ |
| FedNova | UB | ✓ | ✓ | $\mathcal{O}\left(\frac{\sigma^2+G^2}{\sqrt{mR}}+\frac{m(\frac{\sigma^2}{S}+G^2)}{R}\right)$ |
| SCAFFOLD | UB | ✓ | ✗ | $\mathcal{O}\left(\frac{\sigma^2}{S\sqrt{mR}}+\frac{1}{R}\left(\frac{N}{m}\right)^{\frac{2}{3}}\right)$ |
| FedDyn | UB | ✗ | ✗ | $\mathcal{O}(\frac{N}{mR})$ |
| FedPD | ✗ | ✗ | ✓ | $\mathcal{O}\left(\frac{\sigma^2}{S}+\frac{1}{R}+\epsilon\right)$ |
| Our paper FedVRA | AB & TV | ✓ | ✓ & TV | $\mathcal{O}\left(\frac{N\sigma^2}{mS}+\frac{N}{mR}\right)$ |

$R$: Num. of rounds, $m$: Num. of clients sampled, $S$: mini-batch size, $(G, B)$ bounds gradient dissimilarity, $\epsilon$: solution accuracy of local subproblems, UB: unbiased, AB: arbitrary, LSGD: local SGD, TV: time-varying

Table 1: Comparison of representative FL algorithms.

## 1.1 Contributions

In this paper, we are interested in developing FL algorithms that are not only robust against the aforementioned client heterogeneity issues, but also are "adaptive" to the level of them. To this end, we firstly show that the federated ADMM algorithm is in fact a client-variance-reduction scheme, which explains its superior resilience when compared to the primal-only FL algorithms. Inspired by this, we propose a novel primal-dual FL algorithm, termed Federated Variance-Reduction Adaptive (FedVRA), that not only enjoys all the advantages of federated ADMM, but also is adaptive to the degree of client heterogeneity and client sampling schemes. This is achieved by introducing two novel stepsize parameters in the dual variable update and the global variable update, which respectively controls the variance-reduction level and biasness of the global model. Convergence analysis shows that FedVRA can converge to a stationary solution in a sublinear rate under (time-varying) HLU and an arbitrary client sampling scheme, as shown in Table 1. Even intriguingly, we show that FedVRA has an intimate relation with the aforementioned FL algorithms in the sense that they are either special instances of FedVRA or close to it.

Lastly, we extend FedVRA to solve a class of semi/unsupervised problems which involve two blocks of variables and variable constraints. Extensive experiments based on a (semi-)supervised image classification task demonstrate the superiority of FedVRA over the existing schemes.

## 1.2 Notations and Assumptions

Before proceeding, we summarize the key notations in Table 2, and make the following standard assumptions.

**Assumption 1 (Lower boundedness and L-smoothness)** *Each local cost function $f_i(\cdot)$ in problem* (1) *is lower bounded, i.e., $f_i(\mathbf{x}) \geq \underline{f} > -\infty$ and $L_i$-smooth, which implies $\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}')\| \leq L_i\|\mathbf{x}-\mathbf{x}'\|, \forall \mathbf{x}, \mathbf{x}'$.*

| Notation | Definition |
|---|---|
| $\mathcal{A}^r$ | Subset of clients sampled in round $r$ |
| $p_i^r$ | Probability of client $i$ being active in round $r$ |
| $\mathbf{x}_i^{r,t}$ | Variable $\mathbf{x}_i$ of client $i$ at iteration $t$, round $r$ |
| $\mathbf{x}_i^{r+1}$ | Updated variable $\mathbf{x}_i$ of client $i$ in round $r$ |
| $\mathbf{x}_0^r$ | Global variable $\mathbf{x}_0$ at the server in round $r$ |
| $Q_i^r$ | Number of local updates w.r.t. $\mathbf{x}_i$ in round $r$ |
| $\nabla f_i(\mathbf{x}_i;\xi_i)$ | SG of $f_i(\cdot)$ w.r.t $\mathbf{x}_i$ and a sample $\xi_i \in \mathcal{D}_i$ |
| $g_i(\mathbf{x}_i^{r,t})$ | SG of $f_i(\cdot)$ w.r.t $\mathbf{x}_i$ at iteration $t$, round $r$ |
| $\eta_i$ | Stepsize of SGD w.r.t $\mathbf{x}_i$ in round $r$ |
| $\|\cdot\|$ | $l_2$ norm of a vector |
| $[N]$ | Set of $\{1,\ldots,N\}$ |

Table 2: Summary of Notations

**Assumption 2 (Bounded SGD variance)** *For a data sample $\xi_i$ uniformly sampled at random from $\mathcal{D}_i$, the resulting stochastic gradients (SG) for problem* (1) *is unbiased and have bounded variances, i.e.,*

$$\mathbb{E}[\nabla f_i(\mathbf{x}_i;\xi_i)] = \nabla f_i(\mathbf{x}_i), \tag{2}$$

$$\mathbb{E}[\|\nabla f_i(\mathbf{x}_i;\xi_i) - \nabla f_i(\mathbf{x}_i)\|^2] \leq \sigma^2, \tag{3}$$

*where $\sigma > 0$ is a constant.*

Let $g_i(\mathbf{x}_i)$ denote the SG of $f_i(\cdot)$ at $\mathbf{x}_i$ over a mini-batch of $S$ samples i.i.d drawn from the client $i$'s dataset, i.e., $g_i(\mathbf{x}_i) \triangleq \frac{1}{S}\sum_{\xi_i} \nabla f_i(\mathbf{x}_i;\xi_i)$. Then, $g_i(\mathbf{x}_i)$ is unbiased with variance bounded by $\frac{\sigma^2}{S}$.

## 2 Proposed FedVRA Framework

In this section, inspired by the classical distributed ADMM, we propose a unified CVR adaptive FL framework, termed FedVRA, and establish its theoretical property.

### 2.1 Algorithmic Development

Let us start from the classical distributed ADMM method by considering the consensus formulation of problem (1)

$$\min_{\mathbf{x}_0,\mathbf{x}_i,i\in[N]} \sum_{i=1}^{N}\omega_i f_i(\mathbf{x}_i),\ \text{ s.t. } \mathbf{x}_0=\mathbf{x}_i, i\in[N], \tag{4}$$

where $\mathbf{x}_i$ is the local model copy owned by client $i$ and $\mathbf{x}_0$ denotes the global model at the server. Then, we define the corresponding augmented Lagrangian (AL) function

$$\mathcal{L}(\mathbf{x}_0,\{\mathbf{x}_i\},\{\boldsymbol{\lambda}_i\}) \triangleq \sum_{i=1}^{N}\omega_i\mathcal{L}_i(\mathbf{x}_0,\mathbf{x}_i,\boldsymbol{\lambda}_i), \tag{5}$$

$$\mathcal{L}_i(\mathbf{x}_0,\mathbf{x}_i,\boldsymbol{\lambda}_i) \triangleq f_i(\mathbf{x}_i)+\langle\boldsymbol{\lambda}_i,\mathbf{x}_0-\mathbf{x}_i\rangle+\frac{\gamma_i}{2}\|\mathbf{x}_0-\mathbf{x}_i\|^2, \tag{6}$$

where $\boldsymbol{\lambda}_i$ is the Lagrangian dual variable associated with the equality constraint in (4), and $\gamma_i>0$ is the corresponding penalty parameter. The classical ADMM iteratively optimizes the AL function (5) with respect to the variables $\{\mathbf{x}_i,\boldsymbol{\lambda}_i,\mathbf{x}_0\}$ in a Gauss-Seidel fashion. This is, for round $r=0,\ldots,$

$$\mathbf{x}_i^{r+1}=\arg\min_{\mathbf{x}_i}\mathcal{L}_i(\mathbf{x}_0^r,\mathbf{x}_i,\boldsymbol{\lambda}_i^r),\forall i\in[N], \tag{7a}$$

$$\boldsymbol{\lambda}_i^{r+1}=\boldsymbol{\lambda}_i^r+\gamma_i(\mathbf{x}_0^r-\mathbf{x}_i^{r+1}),\forall i\in[N], \tag{7b}$$

$$\begin{aligned}\mathbf{x}_0^{r+1}&=\arg\min_{\mathbf{x}_0}\mathcal{L}(\mathbf{x}_0,\{\mathbf{x}_i^{r+1}\},\{\boldsymbol{\lambda}_i^{r+1}\})\\&=\beta\sum_{i=1}^{N}\omega_i(\gamma_i\mathbf{x}_i^{r+1}-\boldsymbol{\lambda}_i^{r+1}) \qquad (7c)\\&=\mathbf{x}_0^r+\beta\sum_{i=1}^{N}\omega_i\gamma_i(\mathbf{x}_i^{r+1}-\mathbf{x}_0^r)-\beta\sum_{i=1}^{N}\omega_i\boldsymbol{\lambda}_i^{r+1}, \qquad (7d)\end{aligned}$$

where $\beta\triangleq 1/\sum_{i=1}^{N}\omega_i\gamma_i$. As seen, these updates naturally suit for the FL setting as the separable structure of (7a) and (7b) over all pairs $\{(\mathbf{x}_i,\boldsymbol{\lambda}_i)\}$ enables parallel local update of $(\mathbf{x}_i,\boldsymbol{\lambda}_i)$ at the clients while $\mathbf{x}_0$ is updated by the server.

To go beyond the ADMM, like FedAvg, we incorporate the PCP strategy using client sampling and local SGD for handling subproblem (7a). Moreover, we allow time-varying HLU where the clients can perform different numbers of SGD updates in each round. In particular, in round $r$, we let the server sample a small set of $m$ clients $\mathcal{A}^r\subset[N]$ with $\mathrm{Prob}(i\in\mathcal{A}^r)=p_i^r$ and broadcasts $\mathbf{x}_0^r$ to all clients.

- **Local update**: each client $i\in\mathcal{A}^r$ is asked to take $Q_i^r$ consecutive steps of SGD, i.e. $\mathbf{x}_i^{r,0}=\mathbf{x}_0^r,\mathbf{x}_i^{r+1}=\mathbf{x}_i^{r,Q_i^r}$, and $\forall t=0,\ldots,Q_i^r-1$,

$$\mathbf{x}_i^{r,t+1}=\mathbf{x}_i^{r,t}-\eta_i\underbrace{(g_i(\mathbf{x}_i^{r,t})-\boldsymbol{\lambda}_i^r+\gamma_i(\mathbf{x}_i^{r,t}-\mathbf{x}_0^r))}_{\text{SGD of }\mathcal{L}_i\text{ w.r.t }\mathbf{x}_i^{r,t}}, \tag{8}$$

where $\eta_i>0$ is the stepsize. Besides, instead of directly using (7b), we introduce an *adaptive dual stepsize* $a_i^r$ for the dual variable $\boldsymbol{\lambda}_i$, like below

$$\boldsymbol{\lambda}_i^{r+1}=\boldsymbol{\lambda}_i^r+a_i^r\gamma_i(\mathbf{x}_0^r-\mathbf{x}_i^{r+1}),\forall i\in\mathcal{A}^r. \tag{9}$$

The advantage of $a_i^r$ lies in that it enables the fine-grained adaptivity to client heterogeneity as will be discussed in detail in Sec. 2.2. Note that $(\mathbf{x}_i,\boldsymbol{\lambda}_i)$ are unchanged for non-active clients, i.e., $\mathbf{x}_i^{r+1}=\mathbf{x}_0^r,\boldsymbol{\lambda}_i^{r+1}=\boldsymbol{\lambda}_i^r,\forall i\notin\mathcal{A}^r$.

- **Global aggregation**: after receiving $\mathbf{x}_i^{r+1}-\mathbf{x}_0^r$ and $\boldsymbol{\lambda}_i^{r+1},i\in\mathcal{A}^r$, the server aggregates them to produce the new global model $\mathbf{x}_0^{r+1}$ via (7d). Here, we introduce an *aggregation stepsize* $d_i^r$ in (7d) as follows

$$\mathbf{x}_0^{r+1}=\mathbf{x}_0^r+\beta\sum_{i=1}^{N}\omega_i d_i^r\gamma_i(\mathbf{x}_i^{r+1}-\mathbf{x}_0^r)-\beta\sum_{i=1}^{N}\omega_i\boldsymbol{\lambda}_i^{r+1}. \tag{10}$$

The above steps are summarized in Algorithm 1. It is worth noting that, instead of updating $\mathbf{x}_0^{r+1}$ via (10), we split it into two steps in Step 19 and Step 20 of Algorithm 1. By the fact that $\boldsymbol{\lambda}^r=\sum_{i=1}^{N}\omega_i\boldsymbol{\lambda}_i^r,\forall r\geq 0$ and $\mathbf{x}_i^{r+1}=\mathbf{x}_0^r,\forall i\notin\mathcal{A}^r$, one can show that (10) and Step 19 and Step 20 are equivalent. The benefit of doing this splitting is that the client $i$ only needs to upload $a_i^r$ to the server instead of $\boldsymbol{\lambda}_i^{r+1},\forall i\in\mathcal{A}^r$; see Step 15 of Algorithm 1. Therefore, FedVRA has almost the same communication cost per round as FedAvg and does not double it. Besides, we allow HLU in Algorithm 1 by using a time-varying $Q_i^r$ to denote the number of local SGD steps for client $i$ at round $r$. Note that, in practice, the value of $Q_i^r$ depends on the number of local data samples, the mini-batch size and the number of epochs, which are predetermined by the data and computational resources of the client $i$ (Wang et al. 2020; Li et al. 2020a).

Algorithm 1: Proposed FedVRA algorithm

1: **Input:** initial values of $\mathbf{x}_i^0=\mathbf{x}_0$, $\boldsymbol{\lambda}_i^0=\boldsymbol{\lambda}^0=\mathbf{0},\forall i$.
2: **for** round $r=0$ **to** $R-1$ **do**
3: **Server side:** sample clients $\mathcal{A}^r$ from $[N]$ and broadcast $\mathbf{x}_0^r$.
4: **Client side:**
5: **for** client $i=1$ **to** $N$ in parallel **do**
6: **if** client $i\notin\mathcal{A}^r$ **then**
7: Set $\mathbf{x}_i^{r+1}=\mathbf{x}_0^r,\boldsymbol{\lambda}_i^{r+1}=\boldsymbol{\lambda}_i^r$.
8: **else**
9: Set $\mathbf{x}_i^{r,0}=\mathbf{x}_0^r$.
10: **for** $t=0$ **to** $Q_i^r-1$ **do**
11: $\mathbf{x}_i^{r,t+1}=\mathbf{x}_i^{r,t}-\eta_i(g_i(\mathbf{x}_i^{r,t})-\boldsymbol{\lambda}_i^r+\gamma_i(\mathbf{x}_i^{r,t}-\mathbf{x}_0^r))$
12: **end for**
13: Set $\mathbf{x}_i^{r+1}=\mathbf{x}_i^{r,Q_i^r}$.
14: Compute $\boldsymbol{\lambda}_i^{r+1}=\boldsymbol{\lambda}_i^r+a_i^r\gamma_i(\mathbf{x}_0^r-\mathbf{x}_i^{r+1})$.
15: Upload $\gamma_i(\mathbf{x}_i^{r+1}-\mathbf{x}_0^r)$ and $a_i^r$ to the server.
16: **end if**
17: **end for**
18: **Server side:** Compute $\boldsymbol{\lambda}^{r+1}$ and $\mathbf{x}_0^{r+1}$ by
19: $\boldsymbol{\lambda}^{r+1}=\boldsymbol{\lambda}^r+\sum_{i\in\mathcal{A}^r}\omega_i a_i^r\gamma_i(\mathbf{x}_0^r-\mathbf{x}_i^{r+1})$,
20: $\mathbf{x}_0^{r+1}=\mathbf{x}_0^r+\beta\sum_{i\in\mathcal{A}^r}\omega_i d_i^r\gamma_i(\mathbf{x}_i^{r+1}-\mathbf{x}_0^r)-\beta\boldsymbol{\lambda}^{r+1}$.
21: **end for**

## 2.2 ADMM Is a Client-Variance-Reduced Scheme

We remark that the proposed FedVRA algorithm reduces to the federated ADMM when $d_i^r = a_i^r = 1, \forall i, r$ (see (Wang et al. 2022, Section II-B)). As mentioned, it has been found that the distributed ADMM is inherently robust against to data heterogeneity. Here, let us show that federated ADMM is in fact a CVR scheme. To the end, we present the following lemma proved in (Wang et al. 2022, Appendix A).

**Lemma 1** *For any round $r \geq 0$ and client $i \in \mathcal{A}^r$, if $\gamma_i\eta_i \leq 1$, it holds that $\forall t = 0, \ldots, Q_i^r - 1$,*

$$\mathbf{x}_i^{r,t+1} = (1-\gamma_i\eta_i)(\mathbf{x}_i^{r,t} - \widetilde{\eta}_i(g_i(\mathbf{x}_i^{r,t}) + \gamma_i\beta\boldsymbol{\lambda}^r - \boldsymbol{\lambda}_i^r)) + \gamma_i\eta_i\left(\beta\sum_{j=1}^N \omega_j\gamma_j\mathbf{x}_j^r\right), \tag{11}$$

$$\boldsymbol{\lambda}_i^{r+1} = \gamma_i\eta_i\widetilde{Q}_i^r \sum_{t=0}^{Q_i^r-1} \frac{b_i^{r,t}}{\|\mathbf{b}_i^r\|_1} g_i(\mathbf{x}_i^{r,t}) + (1-\gamma_i\eta_i\widetilde{Q}_i^r)\boldsymbol{\lambda}_i^r, \tag{12}$$

*where $\widetilde{\eta}_i \triangleq \frac{\eta_i}{1-\gamma_i\eta_i}$, $\mathbf{b}_i^r \triangleq [b_i^{r,0}, b_i^{r,1}, \ldots, b_i^{r,Q_i^r-1}]^\top \in \mathbb{R}^{Q_i^r}$, $b_i^{r,t} = (1-\gamma_i\eta_i)^{Q_i^r-1-t}$, $\widetilde{Q}_i^r = \|\mathbf{b}_i^r\|_1$, $\boldsymbol{\lambda}^r \triangleq \sum_{i=1}^N \omega_i\boldsymbol{\lambda}_i^r$.*

Impressively, Lemma 1 tells that the federated ADMM is actually a CVR scheme which attempts to reduce the inter-client variance. To elaborate this, firstly by (12), we notice that $\boldsymbol{\lambda}_i^{r+1}$ accumulates the historical *normalized averaging SGs* $\sum_{t=0}^{Q_i^r-1} \frac{b_i^{r,t}}{\|\mathbf{b}_i^r\|_1} g_i(\mathbf{x}_i^{r,t})$. Since $\boldsymbol{\lambda}^r = \sum_{i=1}^N \omega_i\boldsymbol{\lambda}_i^r$, $\boldsymbol{\lambda}^r$ stands for the accumulation of the *global* normalized averaging SG $\sum_{i=1}^N \omega_i \sum_{t=0}^{Q_i^r-1} \frac{b_i^{r,t}}{\|\mathbf{b}_i^r\|_1} g_i(\mathbf{x}_i^{r,t})$. Thus, the term $\gamma_i\beta\boldsymbol{\lambda}^r - \boldsymbol{\lambda}_i^r$ in (11) is actually a *gradient correction* so that $g_i(\mathbf{x}_i^{r,t}) + \gamma_i\beta\boldsymbol{\lambda}^r - \boldsymbol{\lambda}_i^r$ approximates the *global SG*, which shares the same spirit as existing CVR schemes. Furthermore, since $\mathbf{x}_i^{r,t+1}$ in (11) considers the combination with the averaged past model $\beta\sum_{j=1}^N \omega_j\gamma_j\mathbf{x}_j^r$, it can further avoid the local model from deviating from the global one. Therefore, we conclude that federated ADMM is in fact a CVR scheme.

When compared with SCAFFOLD, the proposed FedVRA algorithm not only can adopt time-varying HLU (i.e., using different $Q_i^r$ for different clients and different rounds) but also is more communication efficient since in SCAFFOLD the client needs to upload two vector variables to the server in contrast to one vector variable and one scalar in FedVRA.

## 2.3 Improved Adaptability Beyond ADMM

The introduction of the *adaptive dual stepsize* $a_i^r$ and *aggregation stepsize* $d_i^r$ in FedVRA (Step 14 and Step 19-20 in Algorithm 1) provides two-fold improvements over the vanilla federated ADMM. The first is that the *adaptive dual stepsize* $a_i^r$ enables the algorithm to have extra flexibility in dealing with client heterogeneity, which would accelerate the algorithm convergence. The second is that the algorithm can flexibly control the biasness of the global model for any client sampling scheme.

To see the impact of $a_i^r$, one can follow the same idea as Lemma 1 to show that $\mathbf{x}_i^{r,t+1}$ and $\boldsymbol{\lambda}_i^{r+1}$ of Algorithm 1 satisfy

$$\mathbf{x}_i^{r,t+1} = (1-\gamma_i\eta_i)(\mathbf{x}_i^{r,t} - \widetilde{\eta}_i(g_i(\mathbf{x}_i^{r,t}) + \gamma_i\beta\boldsymbol{\lambda}^r - \boldsymbol{\lambda}_i^r)) + \gamma_i\eta_i\left(\mathbf{x}_0^{r-1} + \beta\sum_{j\in\mathcal{A}^r} \omega_j d_i^r\gamma_j(\mathbf{x}_j^r - \mathbf{x}_0^{r-1})\right), \tag{13}$$

$$\boldsymbol{\lambda}_i^{r+1} = a_i^r\gamma_i\eta_i\widetilde{Q}_i^r G_i^r + (1-a_i^r\gamma_i\eta_i\widetilde{Q}_i^r)\boldsymbol{\lambda}_i^r, \tag{14}$$

where $G_i^r \triangleq \sum_{t=0}^{Q_i^r-1} \frac{b_i^{r,t}}{\|\mathbf{b}_i^r\|_1} g_i(\mathbf{x}_i^{r,t})$. By comparing (14) and (12), one can observe that $a_i^r$ is an independent parameter that controls the weight of current normalized averaging SG $G_i^r$ relative to the historical ones (which are hidden in $\boldsymbol{\lambda}_i^r$). The choice of $a_i^r$ is intimately related to the variance-reduction level. If $a_i^r = 0$, then $\boldsymbol{\lambda}_i^r = \boldsymbol{\lambda}^r = 0$ and the gradient correction term $\gamma_i\beta\boldsymbol{\lambda}^r - \boldsymbol{\lambda}_i^r$ vanishes. Otherwise, a relative large value of $a_i^r$ is preferred when the data distribution gets more non-i.i.d. since fresh $G_i^r$ is more effective in reducing the inter-client variance than the old ones.

The impact of the *aggregation stepsize* $d_i^r$ can be understood as follows. Suppose that $p_i^r = \frac{m}{N}$ for all $i$ and $r$. Denote $\widetilde{\mathbf{x}}_i^{r+1}$ as the local model of client $i$ when it is active in round $r$. Then, by Step 20 of Algorithm 1, we have

$$\begin{aligned}
&\mathbb{E}_{\mathcal{A}^r}\left[\mathbf{x}_0^r + \beta\sum_{i\in\mathcal{A}^r} \omega_i d_i^r\gamma_i(\mathbf{x}_i^{r+1} - \mathbf{x}_0^r) - \beta\boldsymbol{\lambda}^r\right] \\
=&\mathbf{x}_0^r + \beta\frac{m}{N}\sum_{i=1}^N \omega_i d_i^r\gamma_i(\widetilde{\mathbf{x}}_i^{r+1} - \mathbf{x}_0^r) - \beta\boldsymbol{\lambda}^r \qquad (15) \\
=&\begin{cases}
\beta\sum_{i=1}^N \omega_i\gamma_i\widetilde{\mathbf{x}}_i^{r+1} - \beta\boldsymbol{\lambda}^r, \text{ if } d_i^r = \frac{N}{m}, \forall i, r, \\
(1-\frac{m}{N})\mathbf{x}_0^r + \frac{m}{N}\left(\beta\sum_{i=1}^N \omega_i\gamma_i\widetilde{\mathbf{x}}_i^{r+1}\right) - \beta\boldsymbol{\lambda}^r, \text{ if } d_i^r = 1, \forall i, r,
\end{cases} \qquad (16)
\end{aligned}$$

where in (16) two values of $d_i^r = 1$ and $d_i^r = \frac{N}{m}$ are considered. Since $\boldsymbol{\lambda}^r$ is an approximation of the global SG, (15) shows that the expected global model is a gradient descent step. From (16), the starting point is the model average $\sum_{i=1}^N \omega_i\gamma_i\widetilde{\mathbf{x}}_i^{r+1}$ when $d_i^r = \frac{N}{m}$, whereas when, $d_i^r = 1$, the starting point is a convex combination of the model average and the past global model $\mathbf{x}_0^r$. Thereby, choosing a larger value of $d_i^r$ would reduce the bias of the global model and accelerate the algorithm convergence.

## 2.4 Convergence Analysis

The following theorem delineates the convergence conditions for the proposed FedVRA algorithm. The proofs are presented in (Wang et al. 2022, Appendix B and C).

**Theorem 1** *Let $\mathrm{Prob}(i \in \mathcal{A}^r) = p_i^r$ and $0 < \underline{p} \leq p_i^r \leq 1, \forall i, r$. Suppose that the parameters $\gamma_i, a_i^r, \eta_i, d_i^r \forall i, r$, satisfy*

$$\gamma_i \geq \frac{L_i}{2} + \frac{13L_i}{2p_i^r a_i^r\gamma_i\eta_i\widetilde{Q}_i^r}, \tag{17}$$

$$\eta_i \leq \min\left\{\frac{1}{\sqrt{6}\widetilde{Q}_i^r L_i}, \frac{1}{\gamma_i}, \frac{1}{(a_i^r + d_i^r)\gamma_i\widetilde{Q}_i^r}\right\}. \tag{18}$$

*Then, under Assumption 1 and 2, we have*

$$\frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}[\|\nabla f(\mathbf{x}_0^r)\|^2] \le \frac{2D_1(P^0-f)}{\beta R}+\frac{5D_2\sigma^2}{4S}, \quad (19)$$

*where* $D_1 \triangleq \max_r\{\sum_{i=1}^N \frac{9}{p_i^r(2a_i^r+d_i^r)\gamma_i\eta_i\widetilde{Q}_i^r}\}$, $D_2 \triangleq \frac{1}{R}\sum_{r=0}^{R-1}\sum_{i=1}^N \omega_i(D_1(8+2p_i^r(2a_i^r+d_i^r)\gamma_i\eta_i\widetilde{Q}_i^r)+9p_i^r(a_i^r+d_i^r)\gamma_i\eta_i\widetilde{Q}_i^r)$ *and* $P^0 \triangleq \mathbb{E}[f(\mathbf{x}_0^0)]+\sum_{i=1}^N \omega_i\frac{4\beta\mathbb{E}[\|\nabla f_i(\mathbf{x}_0^0)\|^2]}{p_i^0a_i^0\gamma_i\eta_i\widetilde{Q}_i^0}$.

Theorem 1 shows that, if the mini-batch size $S=\sqrt{R}$, then FedVRA converges to a stationary solution in the rate $\mathcal{O}(\frac{1}{R}+\frac{\sigma^2}{\sqrt{R}})$. Since the analysis does not make any assumption on the data homogeneity and client sampling strategy, FedVRA is robust to the non-i.i.d. data distribution and can adapt to arbitrary client sampling schemes. Besides, since the number of local updates $Q_i^r$ can be different for different clients and communication rounds, FedVRA is also robust to time-varying HLU. These aspects are novel when compared to the existing FL algorithms (see Table 1).

It is also observed from Theorem 1 that the convergence of FedVRA is influenced by the constants $D_1$ and $D_2$, which are closely related to the stepsizes $a_i^r$ and $d_i^r$. In particular, if $\gamma_i\eta_i$ is fixed and $p_i^r\gamma_i\eta_i\widetilde{Q}_i^r \ll 1$, then increasing $a_i^r$ or $d_i^r$ properly can potentially decrease both $D_1$ and $D_2$, thereby accelerating the convergence of FedVRA. Such property is consistent with our discussions in the previous subsection and will also be verified through numerical experiments; see Fig. 2 in Sec. 5.1.

When FedVRA adopts a uniform client sampling with $p_i^r=\frac{m}{N}$ for all $i$ and $r$, we have the following corollary.

**Corollary 1** *Given* $p_i^r=\frac{m}{N}, Q_i^r=Q\ge 1, \forall i,r$, *and* $S=\sqrt{R}$, *FedVRA has a convergence rate* $\mathcal{O}(\frac{N}{mR}+\frac{N\sigma^2}{m\sqrt{R}})$.

The proof of Corollary 1 is presented in (Wang et al. 2022, Appendix D). As shown in Table 1, SCAFFOLD seems slightly better in terms of convergence rate, However, FedVRA is more communication-efficient per communication round as discussed in Sec. 2.2. In addition, numerical results suggests that FedVRA has a faster convergence behavior than SCAFFOLD; see Fig. 3 of Sec. 5.1.

**Remark 1** (On the choice of $a_i^r$ and $d_i^r$) Above analysis motivates us to increase the adaptive stepsizes $a_i^r$ and $d_i^r$ to accelerate the algorithm convergence. It would be practically preferred if a particular setup of $(a_i^r, d_i^r)$ is provided. In practice, we suggest choosing these two parameters so that they satisfy $d_i^r=\frac{1}{p_i^r}$ and $(a_i^r+d_i^r)\eta_i^r\gamma_i\tilde{Q}_i^r \le 1$. As explained below (16), the former choice of $d_i^r=\frac{N}{m}$ when $p_i^r=\frac{m}{N}$ reduces the bias of the global model. On the other hand, the latter condition $(a_i^r+d_i^r)\eta_i^r\gamma_i\tilde{Q}_i^r\le 1$ is inspired by the condition of (18). Note that the parameters $(a_i^r, d_i^r)$ chosen in Sec. 5.1 satisfy the conditions, and more importantly, such choice yields much faster convergence and better application performance of FedVRA than its counterparts.

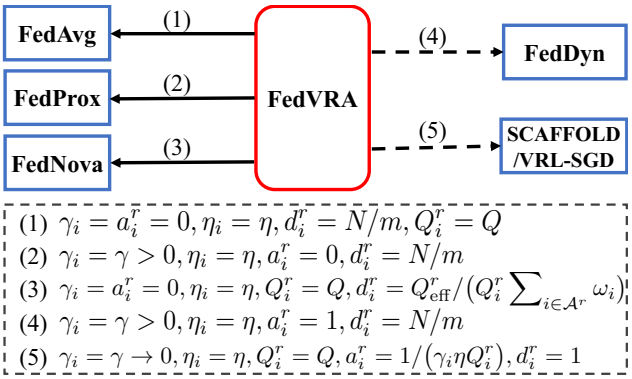


Figure 1: Connections of FedVRA to existing FL algorithms

## 3 Connections to Existing FL Algorithms

Another advantage of the proposed FedVRA algorithm is that it unifies many FL algorithms, including FedAvg, FedProx, FedNova, SCAFFOLD, VRL-SGD, and FedDyn. These algorithms can be exactly recovered or approximated by FedVRA with specific choices of parameters, as illustrated in Fig. 1. This again strengthens our belief of the competitive performance of FedVRA. In particular,

- **FedAvg and FedProx:** By simply setting $a_i^r=0,\eta_i=\eta,d_i^r=\frac{N}{m},Q_i^r=Q$ and $\gamma_i=\gamma=0,\forall i,r$, then $\boldsymbol{\lambda}_i^r=\mathbf{0}$ and $b_i^{r,t}=1,\forall r,t$. The update of FedVRA becomes

$$\mathbf{x}_i^{r,t+1}=\mathbf{x}_i^{r,t}-\eta g_i(\mathbf{x}_i^{r,t}),\forall i\in\mathcal{A}^r, \quad (20)$$

$$\mathbf{x}_0^{r+1}=\mathbf{x}_0^r+\frac{N}{m}\sum_{i\in\mathcal{A}^r}\omega_i(\mathbf{x}_i^{r+1}-\mathbf{x}_0^r). \quad (21)$$

  which exactly recovers FedAvg. The former also reduces to FedProx if we keep $\gamma_i=\gamma>0, i\in[N]$.
- **FedNova:** If we choose $a_i^r=0,\gamma_i=\gamma=0,\eta_i=\eta,d_i^r=\frac{Q_{\text{eff}}^r}{Q_i^r\sum_{i\in\mathcal{A}^r}\omega_i}$, where $Q_{\text{eff}}^r=\frac{\sum_{i\in\mathcal{A}^r}\omega_iQ_i^r}{\sum_{i\in\mathcal{A}^r}\omega_i}$, the update of $\mathbf{x}_0$ in FedVRA can be compactly written as

$$\mathbf{x}_0^{r+1}=\mathbf{x}_0^r-\eta Q_{\text{eff}}^r\sum_{i\in\mathcal{A}^r}\frac{\omega_i}{\sum_{i\in\mathcal{A}^r}\omega_i}\sum_{t=0}^{Q_i^r-1}\frac{g_i(\mathbf{x}_i^{r,t})}{Q_i^r}. \quad (22)$$

  Therefore, such choice ensures the equivalence between FedVRA and FedNova (Wang et al. 2020).
- **SCAFFOLD/VRL-SGD:** The local update of SCAFFOLD is approximated by choosing $\eta_i=\eta,a_i^r=\frac{1}{\gamma_i\eta Q},d_i^r=1,Q_i^r=Q$, and sufficiently small $\gamma_i,\forall i$. To observe this, let us rewrite (11) and (12) with such choice, which gives that, as $\gamma_i=\gamma\to 0,\forall i\in\mathcal{A}^r$,

$$\mathbf{x}_i^{r,t+1}\approx\mathbf{x}_i^{r,t}-\eta(g_i(\mathbf{x}_i^{r,t})+\boldsymbol{\lambda}^r-\boldsymbol{\lambda}_i^r), \quad (23)$$

$$\boldsymbol{\lambda}_i^{r+1}\approx\frac{1}{Q}\sum_{t=0}^{Q-1}g_i(\mathbf{x}_i^{r,t}). \quad (24)$$

  Clearly, the RHS of equations (23) and (24) are identical to the local update of SCAFFOLD. When full participation, equation (23) also reduces to that of VRL-SGD.

- **FedDyn:** FedDyn is related to FedVRA as the former also optimizes the local AL function $\mathcal{L}_i$ in (5) for the local update. However, FedDyn pursues an exact minimizer to (5) for the update of $\mathbf{x}_i^{r+1}$ in place of local SGD. FedVRA approximates FedDyn when we choose $\gamma_i = \gamma > 0, \eta_i = \eta, a_i^r = 1, d_i^r = \frac{N}{m}$.

## 4 Extension to Non-supervised ML Problems

Most FL algorithms are explicitly designed for the smooth problem with one-block of variable, which corresponds to supervised ML tasks. Few (Wang and Chang 2021, 2022) aims to solve the following constrained problem with two blocks of variables, which covers many semi-supervised (Wang et al. 2021b) or unsupervised ML problems (Wang et al. 2021a).

$$\min_{\mathbf{x},\{\mathbf{y}_i\}_i^N} f(\mathbf{x},\mathbf{y}) \triangleq \sum_{i=1}^N \omega_i f_i(\mathbf{x},\mathbf{y}_i) \tag{25a}$$

$$\text{s.t. } \mathbf{x} \in \mathcal{X}, \mathbf{y}_i \in \mathcal{Y}_i, i \in [N], \tag{25b}$$

where $f_i(\cdot,\cdot)$ is local cost function; $\mathcal{X}, \mathcal{Y}_i$ are closed and convex constraint sets. The variable $\mathbf{y}_i$ corresponds to the pseudo labels of client $i$ in semi-supervised FL (Jeong et al. 2021) while it corresponds to the cluster indicators in federated clustering (Wang et al. 2021a). It is crucial to handle problem (25) in FL which, however, is much more challenging to solve than problem (1) because of the extra block of variable $\mathbf{y}_i$ and additional constraint sets $\mathcal{X}, \mathcal{Y}_i$.

We first reformulate problem (25) to the following consensus form and then apply our previous findings.

$$\min_{\substack{\mathbf{x}_i,\mathbf{x}_0,\mathbf{y}_i,\\ i\in[N]}} \sum_{i=1}^N \omega_i f_i(\mathbf{x}_i,\mathbf{y}_i) \tag{26a}$$

$$\text{s.t. } \mathbf{x}_0 \in \mathcal{X}, \mathbf{x}_0 = \mathbf{x}_i, \mathbf{y}_i \in \mathcal{Y}_i, i \in [N]. \tag{26b}$$

Following the same spirit as FedVRA, we present a variant of FedVRA, termed FedVRA-U, to solve problem (26) where the schemes of local SGD and alternating minimization are applied to obtain both the update of $\mathbf{y}_i$ and $\mathbf{x}_i$ at each round. The detailed procedure is summarized in Algorithm 2. Note that $g_i^x(\mathbf{x}_i^{r,t})$ denotes the SG of $f_i(\mathbf{x}_i^{r,t}, \mathbf{y}_i^{r+1})$ w.r.t $\mathbf{x}_i$ at iteration $t$ of round $r$ while $g_i^y(\mathbf{y}_i^{r,t})$ denotes the SG of $f_i(\mathbf{x}_0^r, \mathbf{y}_i^{r,t})$ w.r.t. $\mathbf{y}_i$ at iteration $t$ of round $r$. $\mathcal{P}_{\mathcal{X}}$ and $\mathcal{P}_{\mathcal{Y}_i}$ respectively denote the projection onto the set $\mathcal{X}$ and $\mathcal{Y}_i$. $\eta_i^y$ denotes the stepsize for the update of $\mathbf{y}_i$.

**Convergence analysis:** The following theorem delineates the convergence result of FedVRA-U. The proof is presented in (Wang et al. 2022, Appendix E). Theorem 2 tells that FedVRA-U converges sublinearly to a stationary solution to problem (25). Similar to FedVRA, it is resilient to heterogeneous clients and arbitrary client sampling schemes, which validates the strong scalability of FedVRA to the challenging FL problem (25). The performance of FedVRA-U will be examined later; See Sec. 5.2 for details.

**Theorem 2** *Let* $\text{Prob}(i \in \mathcal{A}^r) = p_i^r$ *and* $0 < \underline{p} \le p_i^r \le \overline{p} < 1, \forall i, r$. *Suppose that the parameters* $\gamma_i, \eta_i, \eta_i^y, a_i^r, d_i^r$ $\forall i, r$, *satisfy*

$$\gamma_i \ge \frac{L_i}{2} + \frac{9L_i}{p_i^r a_i^r \gamma_i \eta_i \widetilde{Q}_i^r \sqrt{1-p_i^r}}, \tag{27}$$

$$\eta_i \le \min\left\{\frac{1}{\sqrt{6}\widetilde{Q}_i^r L_i}, \frac{1}{\gamma_i}, \frac{1}{(a_i^r + d_i^r)\gamma_i \widetilde{Q}_i^r}\right\}, \tag{28}$$

$$\frac{1}{\eta_i^y} \ge L_i + 4\beta Q_{y_i}^r L_i^2 \left(1 + \frac{400}{(a_i^r \gamma_i \eta_i \widetilde{Q}_i^r)^2}\right), \tag{29}$$

*where* $\widetilde{Q}_i^r$ *is defined in Theorem 1, and let the mini-batch size* $S = \sqrt{R}$. *Then, under Assumption 1 and 2, FedVRA-U obtains the convergence rate* $\mathcal{O}(\frac{1}{R} + \frac{\sigma^2}{\sqrt{R}})$.

Algorithm 2: Proposed FedVRA-U algorithm

**Input:** initial values of $\mathbf{x}_0^0 = \mathbf{x}_1^0 = \ldots, \mathbf{x}_N^0, \mathbf{y}_i^0, \boldsymbol{\lambda}_i^0 = \boldsymbol{\lambda}^0 = \mathbf{0}, \forall i$.
**for** round $r = 0$ **to** $R-1$ **do**
  **Server side:** sample clients $\mathcal{A}^r$ from $[N]$ and broadcast $\mathbf{x}_0^r$.
  **Client side:**
  **for** client $i = 1$ **to** $N$ in parallel **do**
    **if** client $i \notin \mathcal{A}^r$ **then**
      Set $\mathbf{x}_i^{r+1} = \mathbf{x}_0^r, \mathbf{y}_i^{r+1} = \mathbf{y}_i^r, \boldsymbol{\lambda}_i^{r+1} = \boldsymbol{\lambda}_i^r$.
    **else**
      Set $\mathbf{y}_i^{r,0} = \mathbf{y}_i^r, \mathbf{x}_i^{r,0} = \mathbf{x}_0^r$.
      **for** epoch $t = 0$ **to** $Q_{y_i}^r - 1$ **do**
        $\mathbf{y}_i^{r,t+1} = \mathcal{P}_{\mathcal{Y}_i}(\mathbf{y}_i^{r,t} - \eta_i^y g_i^y(\mathbf{y}_i^{r,t}))$
      **end for**
      Set $\mathbf{y}_i^{r+1} = \mathbf{y}_i^{r,Q_{y_i}^r}$.
      **for** epoch $t = Q_{y_i}^r$ **to** $\hat{Q}_i^r - 1$ **do**
        $\mathbf{x}_i^{r,t+1} = \mathbf{x}_i^{r,t} - \eta_i(g_i^x(\mathbf{x}_i^{r,t}) - \boldsymbol{\lambda}_i^r + \gamma_i(\mathbf{x}_i^{r,t} - \mathbf{x}_0^r))$
      **end for**
      Set $\mathbf{x}_i^{r+1} = \mathbf{x}_i^{r,\hat{Q}_i^r}$.
      Compute $\boldsymbol{\lambda}_i^{r+1} = \boldsymbol{\lambda}_i^r + a_i^r \gamma_i (\mathbf{x}_0^r - \mathbf{x}_i^{r+1})$.
      Upload $\gamma_i(\mathbf{x}_i^{r+1} - \mathbf{x}_0^r)$ and $a_i^r$ to the server.
    **end if**
  **end for**
  **Server side:** Compute $\boldsymbol{\lambda}^{r+1}$ and $\mathbf{x}_0^{r+1}$ via
  $\boldsymbol{\lambda}^{r+1} = \boldsymbol{\lambda}^r + \sum_{i\in\mathcal{A}^r} \omega_i a_i^r \gamma_i (\mathbf{x}_0^r - \mathbf{x}_i^{r+1})$,
  $\mathbf{x}_0^{r+1} = \mathcal{P}_{\mathcal{X}}\big(\mathbf{x}_0^r + \beta \sum_{i\in\mathcal{A}^r} \omega_i d_i^r \gamma_i (\mathbf{x}_i^{r+1} - \mathbf{x}_0^r) - \beta \boldsymbol{\lambda}^{r+1}\big)$.
**end for**

## 5 Experiment Results

In this section, we will examine the performance of the proposed algorithms by comparing them against four baseline FL algorithms, namely, FedAvg (Li et al. 2020b), FedProx (Li et al. 2020a), SCAFFOLD (Karimireddy et al. 2020), and FedDyn (Acar et al. 2021). All presented results are averaged over 5 runs with different and randomly generated initial points.

**Datasets and models:** The CIFAR-10 (Krizhevsky and Hinton 2009) and MNIST (LeCun, Cortes, and Burges 2010) datasets are considered for evaluation. We respectively adopt a CNN model for the CIFAR-10 and MNIST datasets which are similar to that in (Li et al. 2020b). We simulate the FL process by distributing the training samples of each dataset to

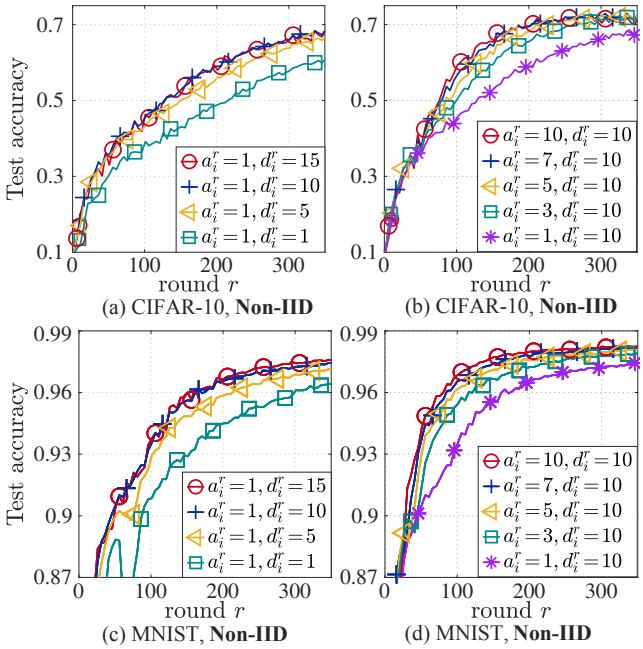


Figure 2: Performance of FedVRA with different $a_i^r$ and $d_i^r$.

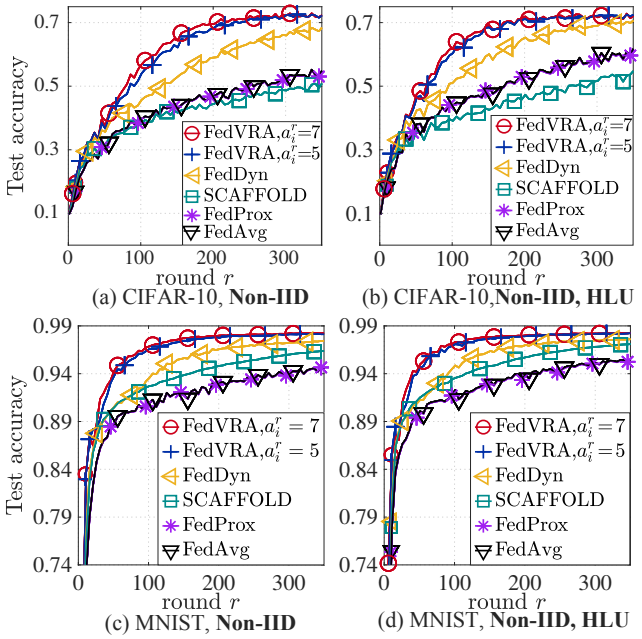


Figure 3: Performance comparison between FedVRA with four FL algorithms. Note that $d_i^r = 10$ for FedVRA.

$N = 100$ clients in two ways: **IID** and **Non-IID**. To obtain **Non-IID**, we follow the partition method as in (Acar et al. 2021) to ensure that $80\%$ of each client's local data samples belong to about 2 classes. The testing samples of each dataset are used to compute test accuracy.

**Parameter setting:** The learning rate $\eta$ is set to be $0.01$ for all algorithms. The mini-batch size $S$ is set to be $50$. In each round, we uniformly sample 10 clients ($|\mathcal{A}^r| = 10$), and randomly choose the number of local epochs from $[1, 5]$ if HLU is considered, and otherwise set it to 2 for each client. Other algorithm specific parameters are tuned individually. FedProx uses $\mu = 0.1$ and SCAFFOLD takes $\eta_g = 1$. Both the parameter $\alpha$ of FedDyn and $\gamma_i$ of FedVRA are $0.1$ by default. All algorithms stop when $500$ rounds are achieved. More details can be found in (Wang et al. 2022, Section V).

## 5.1 Evaluation of Algorithm 1

**Effect of $a_i^r$ and $d_i^r$:** In Fig. 2, we present the performance of FedVRA with different choices of constant $a_i^r$ and $d_i^r$ on both CIFAR-10 and MNIST datasets. One can observe from Fig. (a) and (c) that for constant $a_i^r$, increasing $d_i^r$ can speed up the convergence but may cause some floors. On the contrary, one can see from Fig. (b) and (d) that under **Non-IID** data, increasing $a_i^r$ properly can not only speed up the convergence but also achieve a better test performance. The above results corroborate with our analysis that proper values of $a_i^r$ and $d_i^r$ would boost the convergence of FedVRA. Note that the best choices of $a_i^r$ and $d_i^r$ may depend on the dataset.

**Performance comparison:** In Fig. 3, we compare FedVRA with the four FL algorithms on the non-i.i.d. CIFAR-10 and MNIST datasets. One can see from Fig. 3(b) that on CIFAR-10 dataset, FedVRA significantly outperforms these FL algorithms in terms of both speed and performance. It is worth noting that SCAFFOLD performs comparably to FedAvg under **Non-IID** and even worse than FedAvg if **HLU** is also applied. But our proposed FedVRA algorithm is more resilient to both non-i.i.d. data and HLU, thereby yielding much better performance. We can also observe in Fig. 3(c)(d) the superior performance of FedVRA on the MNIST dataset.

Table 3 and 4 respectively summarize the detailed results, including test accuracy achieved and the number of communication rounds required, on the CIFAR-10 and MNIST datasets. The parameters $d_i^r = 10$ for FedVRA and $a_i^r = 7$ (resp. $a_i^r = 10$) for FedVRA on the CIFAR-10 (resp. MNIST) dataset. One can observe that FedVRA performs the best and achieves much higher test accuracy than FedAvg, FedProx and SCAFFOLD. More importantly, FedVRA is almost 1 time faster than FedDyn and three times faster than the rest. For instance, FedVRA takes $114$ rounds to achieve $\%60$ accuracy on the non-i.i.d. CIFAR-10 dataset while FedDyn needs 206 rounds and the rest requires more than $450$ rounds. The same trend is observed in the case of **IID** and **Non-IID + HLU**. While FedVRA seems slightly more sensitive to different initial points than FedDyn as observed from the 3rd row of Table 3, its faster convergence and better application performance, especially under Non-IID and HLU cases, demonstrates its superior CVR capability over the others. On the MNIST dataset, it can also be observed from Table 4 that FedVRA outperforms the rest in terms of convergence speed, test accuracy achieved and even stability to initial points.

## 5.2 Evaluation of Algorithm 2

We consider the semi-supervised classification task with the same model as in (Wang et al. 2021b) on the CIFAR-10 and MNIST datasets. After data distribution, in each client,

| Cases | Algorithm | Accuracy | #R-50 | #R-60 | #R-70 |
|---|---|---|---|---|---|
| IID | FedAvg | $67.72 \pm 0.20$ | 139 | 282 | $> 500$ |
| | FedProx | $67.62 \pm 0.16$ | 139 | 284 | $> 500$ |
| | SCAFFOLD | $67.32 \pm 0.25$ | 143 | 287 | $> 500$ |
| | FedDyn | $74.53 \pm 0.14$ | 78 | 134 | 250 |
| | FedVRA | $\mathbf{75.47 \pm 0.23}$ | **40** | **67** | **121** |
| Non-IID | FedAvg | $61.03 \pm 1.00$ | 231 | 477 | $> 500$ |
| | FedProx | $61.03 \pm 0.99$ | 231 | 459 | $> 500$ |
| | SCAFFOLD | $55.49 \pm 0.71$ | 318 | $> 500$ | $> 500$ |
| | FedDyn | $71.65 \pm 0.41$ | 130 | 206 | 389 |
| | FedVRA | $\mathbf{73.70 \pm 0.54}$ | **76** | **114** | **191** |
| Non-IID + HLU | FedAvg | $65.20 \pm 1.29$ | 150 | 305 | $> 500$ |
| | FedProx | $65.08 \pm 1.29$ | 157 | 305 | $> 500$ |
| | SCAFFOLD | $59.40 \pm 2.98$ | 260 | 481 | $> 500$ |
| | FedDyn | $72.19 \pm 0.57$ | 95 | 150 | 306 |
| | FedVRA | $\mathbf{73.77 \pm 1.02}$ | **59** | **87** | **159** |

Table 3: Performance comparison between FedVRA and four FL algorithms on the CIFAR-10 dataset. "#R-XX" denotes the number of rounds required to reach XX% accuracy.

| Cases | Algorithm | Accuracy | #R-90 | #R-95 | #R-97 |
|---|---|---|---|---|---|
| Non-IID | FedAvg | $95.64 \pm 0.12$ | 63 | 373 | $> 500$ |
| | FedProx | $95.62 \pm 0.11$ | 63 | 393 | $> 500$ |
| | SCAFFOLD | $97.04 \pm 0.05$ | 44 | 212 | 485 |
| | FedDyn | $97.74 \pm 0.09$ | 42 | 125 | 251 |
| | FedVRA | $\mathbf{98.34 \pm 0.06}$ | **29** | **60** | **107** |
| Non-IID + HLU | FedAvg | $96.18 \pm 0.36$ | 49 | 273 | $> 500$ |
| | FedProx | $96.15 \pm 0.36$ | 49 | 280 | $> 500$ |
| | SCAFFOLD | $97.59 \pm 0.05$ | 33 | 162 | 327 |
| | FedDyn | $97.86 \pm 0.10$ | 34 | 106 | 200 |
| | FedVRA | $\mathbf{98.24 \pm 0.04}$ | **24** | **49** | **87** |

Table 4: Performance comparison between FedVRA and four FL algorithms on the MNIST dataset. "#R-XX" denotes the number of rounds required to reach XX% accuracy.

we randomly select 90% of the local data samples and treat them as the unlabeled data to simulate the semi-supervised FL scenarios where each client only has a few data samples with labels. Then, we examine the performance of FedVRA-U against the four FL algorithm after adapting them to the setting of semi-supervised FL. For FedVRA-U, the parameter $d_i^r = 15$, and the parameters $a_i^r = 7, \gamma_i = 0.1$ (resp. $a_i^r = 5, \gamma_i = 0.5$) on the CIFAR-10 (resp. MNIST) dataset.

Fig. 4 depicts the performance of FedVRA-U and other FL algorithms under different settings. As seen, the proposed FedVRA-U still performs the best and significantly outperform the rest for all cases. Intriguingly, SCAFFOLD performs better than FedDyn, FedProx and FedAvg, but it performs worse than FedVRA-U.

## 6 Conclusion

In this work, inspired by ADMM, we propose an unified client-variance-reduced adaptive FL framework, FedVRA.

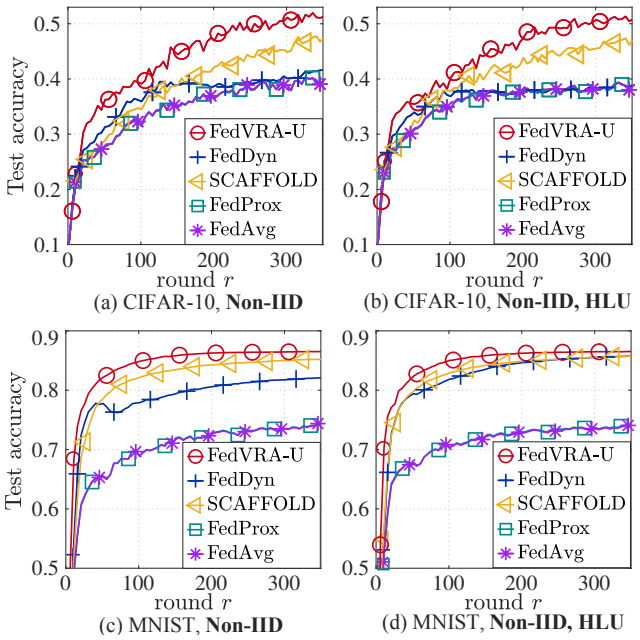


Figure 4: Performance comparison between FedVRA-U with the four FL algorithms.

As a special instance of FedVRA, federated ADMM is a CVR scheme, which explains its inherent robustness to massive heterogeneous clients. FedVRA adopts two novel adaptive stepsizes, which makes it not only retain the ability of CVR but also enjoy better adaptation to the degree of client heterogeneity and client sampling schemes. In addition, we spotlight that FedVRA unifies many representative FL algorithms. Its superior performance is validated both theoretically and empirically.

## Acknowledgements

The work of Shuai Wang and Tony Q. S. Quek was supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research & Development Programme. The work of Tsung-Hui Chang was supported in part by Shenzhen Science and Technology Program under Grant No. JCYJ20190813171003723 and RCJC20210609104448114, in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2021B1515120008, and in part by Guangdong Provincial Key Laboratory of Big Data Computing. The work of Yanqing Xu was supported by the NSFC, China, under Grant 62201486. The work of Zhiguo Wang was supported in part by the NSFC, China, under Grant 62203313, and in part by the Fundamental Research Funds for the Central Universities and Natural Science Foundation of Sichuan Province under Grant 2022NSFSC1853. The work of Defeng Sun was supported by the Hong Kong Research Grant Council under Grant 15303720.

## References

Acar, D. A. E.; Zhao, Y.; Matas, R.; Mattina, M.; Whatmough, P.; and Saligrama, V. 2021. Federated Learning Based on

Dynamic Regularization. In *Proc. ICLR*, 1–6. Virtual Conference.

Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; and Eckstein, J. 2010. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3, no. 1: 1–122.

Chang, T.-H.; Hong, M.; Wai, H.-T.; Zhang, X.; and Lu, S. 2020. Distributed Learning in the Nonconvex World: From batch data to streaming and beyond. *IEEE Signal Process. Mag.*, 37(3): 26–38.

Cho, Y. J.; Wang, J.; and Joshi, G. 2022. Client Selection in Federated Learning: Convergence Analysis and Power-of-Choice Selection Strategies. In *Proc. AISTATS*, 1369–1375. Virtual Conference.

Gong, Y.; Li, Y.; and Freris, N. M. 2022. FedADMM: A Robust Federated Deep Learning Framework with Adaptivity to System Heterogeneity. In *Proc. ICDE*, 1–12. Virtual Conference.

Hajinezhad, D.; Hong, M.; Zhao, T.; and Wang, Z. 2016. NESTT: A Nonconvex Primal-Dual Splitting Method for Distributed and Stochastic Optimization. In *Proc. NeurIPS*, 3215–3223. Barcelona, SPAIN.

Jeong, W.; Yoon, J.; Yang, E. S.; and Hwang, J. 2021. Federated Semi-Supervised Learing with Inter-Client Consistency and Disjoint learning. In *Proc. ICLR*, 1–7. Virtual Conference.

Karimireddy, S. P.; Kale, S.; Mohri, M.; Reddi, S.; Stich, S.; and Suresh, A. T. 2020. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In *Proc. ICML*, 5132–5143.

Krizhevsky, A.; and Hinton, G. 2009. Learning Multiple Layers of Features from Tiny Images. *Master's thesis, Department of Computer Science, University of Toronto*.

Kŏnecný, J.; McMahan, H. B.; and Ramage, D. 2015. Federated Optimization: Distributed Optimization Beyond the Datacenter. In *NeuIPS Optimization for Machine Learning Workshop*, 1–5. Montreal, Quebec, Canada.

Kŏnecný, J.; McMahan, H. B.; Ramage, D.; and Richtarik, P. 2016. Federated Optimization: Distributed Machine Learning for On-Device Intelligence. *arXiv preprint arXiv:1610.02527*.

LeCun, Y.; Cortes, C.; and Burges, C. 2010. MNIST Handwritten Digit Database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

Li, T.; Sahu, A. K.; Sanjabi, M.; Zaheer, M.; Talwalkar, A.; and Smith, V. 2020a. Federated Optimization in Heterogeneous Networks. In *Proc. MLSys*, 1–12. Austin, TX, USA.

Li, X.; Huang, K.; Yang, W.; Wang, S.; and Zhang, Z. 2020b. On the Convergence of FedAvg on Non-IID Data. In *Proc. ICLR*, 1–11. Addis Ababa, ETHIOPIA.

Liang, X.; Shen, S.; Liu, J.; Pan, Z.; Chen, E.; and Cheng, Y. 2020. Variance Reduced Local SGD with Lower Communication Complexity. *arXiv preprint arXiv:1912.12844*.

McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; and Areas, B. A. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proc. ICML*, 1–10. Sydney, Australia.

Stich, S. U. 2019. Local SGD Converges Fast and Communicates Little. In *Proc. ICLR*, 1–5. New Orleans, LA, USA.

Wang, J.; Liu, Q.; Liang, H.; Joshi, G.; and Poor, H. V. 2020. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. In *Proc. NeurIPS*, 1–13. Virtual Conference.

Wang, S.; and Chang, T.-H. 2021. Demystifying Model Averaging for Communication-Efficient Federated Matrix Factorization. In *Proc. IEEE ICASSP*, 3680–3684. Toronto, Ontario, Canada.

Wang, S.; and Chang, T.-H. 2022. Federated Matrix Factorization: Algorithm Design and Application to Data Clustering. *IEEE Transactions on Signal Processing*, 70: 1625–1640.

Wang, S.; Chang, T.-H.; Cui, Y.; and Pang, J.-S. 2021a. Clustering by Orthogonal NMF Model and Non-Convex Penalty Optimization. *IEEE Transactions on Signal Processing*, 69: 5273–5288.

Wang, S.; Xu, Y.; Wang, Z.; Chang, T.-H.; Quek, T. Q. S.; and Sun, D. 2022. Supplementary Material of Beyond ADMM: A Unified Client-variance-reduced Adaptive Federated Learning Framework. *arXiv preprint arXiv:2212.01519*.

Wang, Z.; Wang, X.; Sun, R.; and Chang, T.-H. 2021b. Federated Semi-Supervised Learning with Class Distribution Mismatch. *arXiv preprint arXiv: 2111.00010*.

Yang, Q.; Liu, Y.; Chen, T.; and Tong, Y. 2019. Federated Machine Learning: Concept and Applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2): 1–19.

Zhang, X.; Hong, M.; Dhople, S.; Yin, W.; and Liu, Y. 2021. FedPD: A Federated Learning Framework with Adaptivity to Non-IID Data. *IEEE Transactions on Signal Processing*, 1(1): 1–15.