# FEDGSNR: ACCELERATING FEDERATED LEARNING ON NON-IID DATA VIA MAXIMUM GRADIENT SIGNAL TO NOISE RATIO

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Federated learning (FL) allows participants jointly training a model without direct data sharing. In such a process, participants rather than the central server perform local updates of stochastic gradient descent (SGD) and the central server aggregates the gradients from the participants to update the global model. However, the non-iid training data in participants significantly impact global model convergence. Most of existing studies addressed this issue by utilizing variance reduction or regularization. However, these studies focusing on specific datasets lack theoretical guarantee for efficient model training. In this paper, we provide a novel perspective on the non-iid issue by optimizing Gradient Signal to Noise Ratio (GSNR) during model training. In each participant, we decompose local gradients calculated on the non-iid training data into the signal and noise components and then speed up the model convergence by maximizing GSNR. We prove that GSNR can be maximized by using the optimal number of local updates. Subsequently, we develop FedGSNR to compute the optimal number of local updates for each participant, which can be applied to existing gradient calculation algorithms to accelerate the global model convergence. Moreover, according to the positive correlation between GSNR and the quality of shared information, FedGSNR allows the server to accurately evaluate contributions of different participants (i.e., the quality of local datasets) by utilizing GSNR. Extensive experimental evaluations demonstrate that FedGSNR achieves on average a $1.69\times$ speedup with comparable accuracy.

## 1 INTRODUCTION

Federated learning (FL) McMahan et al. (2017) focuses on a practical scenario with multiple participants to collaboratively train a model without direct data sharing. Different from the typical centralized optimization problem, FL decomposes the optimization problem into several sub-optimization problems, and distributes them to different participants to be solved separately with the corresponding local datasets. Moreover, these local datasets often follow non-iid distributions in reality. During the training phase, each participant solves the sub-problem via stochastic gradient decent (SGD), and sends back the corresponding results for aggregation. One of the most popular FL algorithms is FedAvg McMahan et al. (2017), and it typically accelerates global model convergence through multiple local updates. Although it has shown great performance in many practical applications, there're still mysteries in this area, especially in non-iid cases, and many previous literatures Haddadpour & Mahdavi (2019); Khaled et al. (2020); Li et al. (2020b) make efforts to analyze the convergency or even to accelerate it.

One of the key challenges in FL is how a model can be well trained on non-iid data in different participants. On the one hand, such imbalance breaks the unbiased optimization procedure when we utilize multiple local updates. While on the other hand, due to the differences between different local datasets, non-iid information distribution makes it difficult to evaluate the contribution of different participants. The former slows down the FL-based model convergence, i.e., a key factor of the efficiency. The latter is associated with contribution evaluation involving malicious data tampering detection, contribution-based profit distribution, incentive mechanism design, etc. Especially in the current data-driven age Sim et al. (2020), contribution evaluation is particularly important. In addition

to separately pursuing the two goals of accelerating model convergence and improving contribution evaluation accuracy, these two goals even conflict with each other.

To solve the above challenges, we propose a novel approach that speedups model training by maximizing Gradient Signal to Noise Ratio (GSNR). The intuitions behind the design are two-fold. First, there is always a global optimal solution no matter how the data is distributed. For each local dataset, we can obtain an optimal optimization direction, i.e., the global gradient. Second, based on the information theory, GSNR determines the channel capacity and a larger GSNR can accelerate the model convergence. Thus, we can decompose the local optimization direction (i.e., the local gradient) into mutually orthogonal signal vector and noise vector. We find that if we can obtain the global gradient, the signal vector is parallel to the global gradient, while the noise vector is orthogonal to it. For example, Fig. 1 shows a typical example of orthogonal decomposition of two participants.

We prove that the number of local updates can control the GSNR value and we can maximize GSNR by computing the optimal number of local updates. To maximize GSNR, we utilize the gradients uploaded by the participants to estimate the global gradient, and propose a FedGSNR algorithm to compute the optimal number of local updates according to the estimated the global gradient. Moreover, based on the GSNR perspective, we also develop specific methods to compute the GSRN for each dataset, which allow the server to evaluate each participant's contribution.

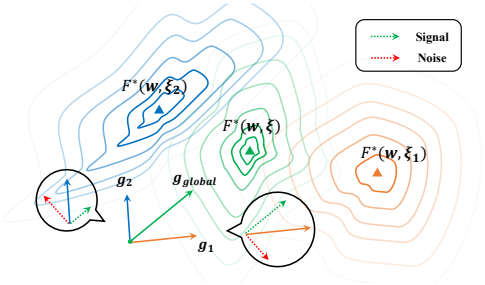



Figure 1: An example of Gradient Signal to Noise Ratio (GSNR). A local step can be decomposed into two components: signal and noise, the former parallels to the update with global data, and the latter orthogonal to it.

In addition to personalizing the number of local updates to optimize model convergence efficiency, the newly proposed GSNR strategy FedGSNR is orthogonal to existing methods, which mostly depends on modifying gradients calculating. Hence, FedGSNR can be combined with these methods so as to further improve them.

In summary, our contributions in this paper are as follows:

- We prove that the optimal local updates decides the maximal GSNR, which leads to faster and more stable convergence.
- We analyze existing FL algorithms with the perspective of GSNR. Moreover, based on the viewpoint of GSNR maximization, we propose an algorithm FedGSNR, which can be combined with most of current FL algorithms to calculate its optimal local updates.
- We derive a function $r(\boldsymbol{w})$ to calculate GSNR, which can be utilized to evaluate the local contributions of different participants.
- We confirm our theoretical results on CIFAR-10 and CIFAR-100 datasets, and experiments indicate that FedGSNR can achieve on average a 1.69$\times$ speedup over its original when the information unevenly distributed among all participants, and $r(\boldsymbol{w})$ is a reasonable metric for local contributions.

**Related Work.** There has been a lot of literatures devoted to improving FL, including convergence Karimireddy et al. (2020); Li et al. (2020b); Wang et al. (2020a); Reddi et al. (2021) robustness Mohri et al. (2019); Fang et al. (2020); Li et al. (2021), and data privacy Melis et al. (2019); Zhu et al. (2019); Bagdasaryan et al. (2018). Regarding GSNR, Rainforth et al. (2018); Liu et al. (2020) try to analyze the generalization and variational bounds with such a concept. In this work, we focus on the relationship between GSNR and the optimal local updates in FL scenarios. To control the noise component (client drift), Karimireddy et al. (2020) proposes a specific gradient calculating method based on variance reduction. Li et al. (2020a) indicates that under non-iid FL conditions, a large number of local updates lead to divergence or instability. While Wang et al. (2020b) tries to stabilize the training procedure with a new average strategy. On the other hand, Wang et al. (2019) proposes a practical optimization problem with resources constraints, and it determines the number of local updates for each participant according to the corresponding constraints.

A similar work Khaled et al. (2020) derives an upper bound of local updates by total iterations $T$ and the number of participants $M$, which proposes a theoretical analysis of local updates. But, they treat each participant equally, and fail to propose a method to calculate the optimal number of local updates directly from the heterogeneous data. More discussion of related work can be found in Appendix A.

## 2 PRELIMINARY

### 2.1 FEDERATED AVERAGING (FEDAVG)

In this work, we consider the following federated optimization problem:

$$\min_{\boldsymbol{w}} \overline{F(\boldsymbol{w})} := \mathbb{E}_{\boldsymbol{\xi}}[F(\boldsymbol{w},\ \boldsymbol{\xi})] = \mathbb{E}_{C}[\mathbb{E}_{\boldsymbol{\xi}}[F(\boldsymbol{w},\ \boldsymbol{\xi})] \mid C] = \sum_{k=1}^{K} P(C = C_k) \cdot \mathbb{E}_{\boldsymbol{\xi}}[F(\boldsymbol{w},\ \boldsymbol{\xi}|C_k)], \quad (1)$$

where $F(\boldsymbol{w},\ \cdot\ )$ is a specified loss function with model $\boldsymbol{w}$, $K$ is the number of participants, and $P(C)$ is a discrete probability distribution correlated to the importance of different datasets. Usually, $P(C)$ is a uniform distribution or proportional to the local data quantity, and $\boldsymbol{\xi}|C_k$ is a random sample drawn from the dataset of $k$-th participant, i.e., $\boldsymbol{\xi}|C_k \sim p(\boldsymbol{x}|C_k)$.

Regarding traditional machine learning, the global dataset is gathered from all participants, and the goal is to minimize

$$F(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{\xi}}[F(\boldsymbol{w},\ \boldsymbol{\xi})], \quad (2)$$

where $\boldsymbol{\xi}$ is a random sample of global dataset, i.e., the gathered data. However, in most cases, due to the privacy requests, we cannot gather data from different participants. Thus, we separate the target function as Eq. (1), and send the initial model $\boldsymbol{w}_1$ to each participant, then they do the optimization locally, and send back the corresponding results. We finally obtain the results by Eq. (1).

If each participant does only one step optimization, according to the property of conditional expectation, minimizing Eq. (1) is equivalent to minimizing Eq. (2). But this procedure puts a lot of pressure on communication, so researchers propose to do more local updates for efficiency. Hence, for the $k$-th participant, the optimization procedure of a typical round can be formalized as

$$\boldsymbol{w}_{i+1}^{k} \leftarrow \boldsymbol{w}_{i}^{k} - \eta \mathbb{E}_{\boldsymbol{\xi}|C_k}[\nabla_{\boldsymbol{w}} F(\boldsymbol{w}_{i}^{k}, \boldsymbol{\xi}|C_k)],\ i = 1, \cdots, n.$$

Then the central server aggregates local models $\boldsymbol{w}_{n+1}^{1}, \cdots, \boldsymbol{w}_{n+1}^{K}$ to update the global model by

$$\overline{\boldsymbol{w}} = \sum_{k=1}^{K} p_k \boldsymbol{w}_{n+1}^{k}, \quad (3)$$

where we denote $p_k$ for $P(C = C_k)$ for convenience.

### 2.2 WASSERSTEIN DISTANCE

Wasserstein distance (Villani (2009)) is a metric in probabilistic space inspired by the problem of optimal transport. It is a distance between probability distributions that takes geometric information into account. The general wasserstein distance is defined as

$$\boldsymbol{W}_p(\boldsymbol{\mu}, \boldsymbol{\nu}) = \inf_{\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(\boldsymbol{\mu}, \boldsymbol{\nu})} \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{y}) \sim \boldsymbol{\gamma}}[\|\boldsymbol{x} - \boldsymbol{y}\|_p],$$

which is difficult to find a closed form solution. However, if we chose 2-norm as the geometric measure and simplify the problem to Gaussian distribution, the distance has an analytic solution

$$d^2 = \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2 + tr((\boldsymbol{\Sigma}_1^{\frac{1}{2}} - \boldsymbol{\Sigma}_2^{\frac{1}{2}})^2), \quad (4)$$

where we define $d := \boldsymbol{W}_2(\mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2))$.

## 3 MAXIMIZE GSNR WITH OPTIMAL LOCAL UPDATES

In this section, we investigate the relationship between GSNR and the local updates, then we propose a method to calculate the optimal number of local updates. As Section 4 will introduce, GSNR can be

calculated by the ratio between the norm of global gradient, which is a constant for all participants, and the distance between global and local gradients. Therefore, to maximize GSNR is equivalent to minimize the distance between global and local gradients. Furthermore, the minimal distance, i.e., the maximum GSNR, is decided by the optimal local updates.

On the other hand, as mentioned in Section 2.1, our target is to gather data so as to optimize Eq. (2) centralizedly. But in practice, due to some real restrictions, we can just optimize Eq. (1) distributedly. Thus, we treat the former procedure as an ideal optimization process. Based upon this idea, for accelerating convergence, our practical optimization problem also requires to minimize the distance between practical and ideal optimization path, which is determined by the corresponding gradients.

The distance between practical and ideal optimization path can be formalized as

$$D = \boldsymbol{W}_2(p_{\overline{\boldsymbol{w}}}(\boldsymbol{w}), p_{\boldsymbol{w}_g}(\boldsymbol{w})), \tag{5}$$

where $\boldsymbol{w}_g$ is the ideal optimization path based on the global data distribution, i.e., the distribution of data gathered from all participants, while $\overline{\boldsymbol{w}}$ is the corresponding practical path defined by Eq. (3). According to Eq. (3), due to the conditional independence of the data from different participants, the random vector $\overline{\boldsymbol{w}}$ is a convex combination of a set of independent random vectors, hence $p_{\overline{\boldsymbol{w}}}(\boldsymbol{w})$ is identified by the convolution of corresponding local distribution functions, and such a distribution contains all details of each participant. Intuitively, if a specific participant attempts to minimize Eq. (5), it has to gather all information from others, which violates the privacy requests. Then we need to derive an upper bound of Eq. (5) as

$$\begin{aligned} D = \boldsymbol{W}_2(p_{\overline{\boldsymbol{w}}}(\boldsymbol{w}), p_{\boldsymbol{w}_g}(\boldsymbol{w})) &= \inf_{\boldsymbol{\gamma} \in \boldsymbol{\Gamma}(p_{\overline{\boldsymbol{w}}}, p_{\boldsymbol{w}_g})} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}) \sim \boldsymbol{\gamma}}[\|\boldsymbol{x} - \boldsymbol{y}\|_2] \\ &\leq \sum_{k=1}^{K} p_k \inf_{\boldsymbol{\gamma_k} \in \boldsymbol{\Gamma}(p_{\boldsymbol{w}_{n+1}^k}, p_{\boldsymbol{w}_g})} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}) \sim \boldsymbol{\gamma_k}}[\|\boldsymbol{x} - \boldsymbol{y}\|_2], \end{aligned} \tag{6}$$

where we split $\|\boldsymbol{x} - \boldsymbol{y}\|_2$ as $\|\sum_{k=1}^{K} p_k(\boldsymbol{x}_k - \boldsymbol{y})\|_2$, and each pair $(\boldsymbol{x}_k, \boldsymbol{y})$ is supported on $\boldsymbol{\Gamma}(p_{\boldsymbol{w}_{n+1}^k}, p_{\boldsymbol{w}_g})$. Then the inequality depends on triangle inequality and the fact that $\boldsymbol{w}_{n+1}^k$ is mutual independent. The upper bound is obvious, since by optimal transport, comprehensively considering all mounds is better than the sum of separate consideration.

By upper bound (6), the distance between practical and ideal optimization path is upper bounded. In other words, while minimizing upper bound (6), the target distance (5) is approximately minimized. Specifically, the target gap vanishes as the upper bound approaches to 0. Regarding Eq. (6), it is the sum of the distance between each independent local distribution $p_{\boldsymbol{w}_{n+1}^k}$ and the global distribution $p_{\boldsymbol{w}_g}$, then due to the rotating symmetry, we can minimize upper bound (6) separately for each participant. Hence, without loss of generality, we just consider a specific participant in federated learning in the rest of this paper.

Based on former analysis, for maximizing GSNR, each participant needs to greedily optimize its local gradient distribution to minimize the distance $\boldsymbol{W}_2(p_{\boldsymbol{w}_{n+1}^k}(\boldsymbol{w}), p_{\boldsymbol{w}_g}(\boldsymbol{w}))$. As the initial parameters, i.e., $\boldsymbol{w}_1^k$, for all participants are identical, the main target is to estimate the gradient distribution of different participants. Then we have the following assumption.

**Assumption 3.1. (Bounded variance)** The variance of stochastic gradients are uniformly bounded, i.e., $\mathbb{E}_{\boldsymbol{\xi}|C_i}\|\nabla_{\boldsymbol{w}} F(\boldsymbol{w}, \boldsymbol{\xi}|C_i) - \boldsymbol{\mu}_i\|^2 \leq \sigma^2,\ \forall i,\ \boldsymbol{w}$, where $\boldsymbol{\mu}_i := \mathbb{E}_{\boldsymbol{\xi}|C_i}[\nabla_{\boldsymbol{w}} F(\boldsymbol{w}, \boldsymbol{\xi}|C_i)]$.

Under such an assumption, the mini-batch stochastic gradient descent converges to joint normal distribution. The detailed proofs can be found in Appendix B.

**Lemma 3.2.** *With Assumption 3.1, let $\boldsymbol{\xi}_{i,b}|C_k$ be a sample from $k$-th local dataset uniformly at random, $\boldsymbol{g} = (\boldsymbol{g}_1, \boldsymbol{g}_2, \cdots, \boldsymbol{g}_n)$ be a finite dimensional gradient vector, where $\boldsymbol{g}_i = \frac{1}{B}\sum_{b=1}^{B} \nabla_{\boldsymbol{w}} F(\boldsymbol{w}_i, \boldsymbol{\xi}_{i,b}|C_k),\ i \in \{1, \cdots, n\}$, then $\sqrt{B}(\boldsymbol{g} - \mathbb{E}[\boldsymbol{g}])$ converges to normal distribution.*

Lemma 3.2 implies that, with mini-batch stochastic gradient descent, the sum of local updates converges to a Gaussian distribution, i.e., $\bar{\boldsymbol{g}} = \boldsymbol{w}_{n+1} - \boldsymbol{w}_1 = \sum_{i=1}^{n} \boldsymbol{g}_i = \mathbb{1}^T \boldsymbol{g}$, where $\bar{\boldsymbol{g}}$ is a linear transformation of a joint Gaussian vector. Thus, with a finite batch size $B$, it can be approximated by Gaussian distribution. Then we need to calculate the mean vector and the covariance matrix. For such a purpose, we have another assumption of smoothness.

**Assumption 3.3. (Smoothness)** The target function $F(\boldsymbol{w}, \cdot): R^m \to R$ is twice differentiable, and the expected matrix norm of hessian matrix $H(F(\boldsymbol{w}, \cdot))$ is bounded, i.e., $\mathbb{E}_{\boldsymbol{\xi}}\|H(F(\boldsymbol{w}, \boldsymbol{\xi}))\|^2 \leq L^2$, where $\boldsymbol{\xi}$ is randomly sampled from a specific dataset.

Note that Assumption 3.3 is weaker than L-smooth Assumption, since if a function $F(\boldsymbol{w}, \cdot)$ is L-smooth, it conforms to Assumption 3.3, but not vice versa.

Assumption 3.3 always holds for typical machine learning tasks, e.g., logistic regression, soft-max classification and so on. With these assumptions, we have the following lemma.

**Lemma 3.4.** *If Assumption 3.1 and 3.3 hold, let $\{\eta_r\}_{r=1}^{+\infty}$ be a sequence of real number such that $\lim_{r\to+\infty} \eta_r = 0$, and $\{\boldsymbol{\varepsilon}_r\}_{r=1}^{+\infty}$ be a sequence of random vectors, where $\boldsymbol{\varepsilon}_r = \hat{\boldsymbol{g}} - \bar{\boldsymbol{g}}$, and $\hat{\boldsymbol{g}} = \sum_{i=1}^{n} \nabla_{\boldsymbol{w}} F(\boldsymbol{w}_1, \boldsymbol{\xi}_i)$, $\bar{\boldsymbol{g}} = \sum_{i=1}^{n} \nabla_{\boldsymbol{w}} F(\boldsymbol{w}_i, \boldsymbol{\xi}_i)$ with $\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \eta_r \boldsymbol{g}_{i-1}, i \in \{2, \cdots, n\}$ respectively, then we have $\boldsymbol{\varepsilon}_r \xrightarrow{L} 0$, which implies $\hat{\boldsymbol{g}} \xrightarrow{L} \bar{\boldsymbol{g}}$.*

Lemma 3.4 implies that if we introduce learning rate decay[1], then

$$\forall \epsilon, \lim_{r\to+\infty} Pr(\|\bar{\boldsymbol{g}} - \hat{\boldsymbol{g}}\| > \epsilon) = 0. \tag{7}$$

In a typical communication round $r$, the optimization process implies that $\boldsymbol{w}_{n+1} - \boldsymbol{w}_1 = \eta_r \bar{\boldsymbol{g}}$. While based on Eq. (7), we can use $\eta_r \hat{\boldsymbol{g}} = \eta_r \sum_{i=1}^{n} \nabla_{\boldsymbol{w}} F(\boldsymbol{w}_1, \boldsymbol{\xi}_i)$ to estimate $\eta_r \bar{\boldsymbol{g}}$.

Specifically, since $\boldsymbol{w}_1$ is a constant vector and $\boldsymbol{\xi}_i$ is iid sampled from the dataset in different local steps, $\{\nabla_{\boldsymbol{w}} F(\boldsymbol{w}_1, \boldsymbol{\xi}_i) | i \in \{1, \cdots, n\}\}$ are also iid random vectors. Therefore, based on the sum of independent random variables, we have $\boldsymbol{\mu} = \mathbb{E}[\eta_r \hat{\boldsymbol{g}}] = n\eta_r \mathbb{E}_{\boldsymbol{\xi}}[F(\boldsymbol{w}_1, \boldsymbol{\xi})]$, and $\boldsymbol{\Sigma} = Cov(\eta_r \hat{\boldsymbol{g}}, \eta_r \hat{\boldsymbol{g}}) = n\eta_r^2 \boldsymbol{\Sigma}_1$.

Additionally, $\boldsymbol{\Sigma}$ is a second order term. To simplify the analysis, we need to convert coefficient $n$ to $n^2$. As mentioned before, $\boldsymbol{\Sigma}_1$ is the covariance matrix of a mean vector distribution, hence it depends on the batch size $B$. Let $B = \hat{B}/n$, then we obtain $\boldsymbol{\Sigma} = n^2 \eta^2 \boldsymbol{\Sigma}_1 / \hat{B}$.

For convenience, in the rest of our work, we use $\boldsymbol{\mu}_*$ and $\boldsymbol{\Sigma}_*$ to denote the corresponding mean vector and covariance matrix of gradients estimated by a specific dataset. Similarly, as we can change the local batch size for simplifying computation, we ignore the differences between $B$ and $\hat{B}$. Then the optimal number of local updates is implied by following theorem.

**Theorem 3.5.** *With Assumption 3.1 and 3.3, for one global update, the optimal number of local updates is*

$$n_1^{opt} = \max(0, \frac{\boldsymbol{\mu}_l^T \boldsymbol{\mu}_g + \frac{tr((\boldsymbol{\Sigma}_l \boldsymbol{\Sigma}_g)^{\frac{1}{2}})}{B}}{\|\boldsymbol{\mu}_l\|^2 + \frac{tr(\boldsymbol{\Sigma}_l)}{B}}),$$

*and the optimal distance is $(\Delta_1^{opt})^2 = \eta_r^2 \Delta^2$,*

$$\Delta^2 = \|\boldsymbol{\mu}_g\|^2 + \frac{tr(\boldsymbol{\Sigma}_g)}{B} - \frac{\left(\boldsymbol{\mu}_l^T \boldsymbol{\mu}_g + \frac{tr((\boldsymbol{\Sigma}_l \boldsymbol{\Sigma}_g)^{\frac{1}{2}})}{B}\right)^2}{\|\boldsymbol{\mu}_l\|^2 + \frac{tr(\boldsymbol{\Sigma}_l)}{B}},$$

*where $\boldsymbol{\mu}_l$, $\boldsymbol{\Sigma}_l$ represent mean vector and covariance matrix of gradients estimated by local dataset with initial value $\boldsymbol{w}_1$, while $\boldsymbol{\mu}_g$, $\boldsymbol{\Sigma}_g$ represent the corresponding global statistics.*

**Corollary 3.6.** *For $m$ global updates, where $m$ is a constant, the optimal number of local updates to minimize the distance is $n_m^{opt} = m \cdot n_1^{opt}$ and the optimal distance is $(\Delta_m^{opt})^2 = m^2 \cdot (\Delta_1^{opt})^2$.*

On the one hand, according to Theorem 3.5 and Corollary 3.6, the optimal local updates is independent of the learning rate $\eta_r$, and it is decided by the differences between global and local gradient distribution. On the other hand, the distance between ideal path and the practical path is correlated to $\eta_r$. Hence, with a sufficient small $\eta_r$, the gap vanishes. That's the reason why existing literatures argue FedAvg with a small learning rate is equivalent to stochastic gradient descent, and the diminishing learning rate is crucial for the convergence of FL Khaled et al. (2019); Li et al. (2020b).

[1]For example, $\eta_r = \eta_0 \alpha^r$ refers to a widely used learning rate decay method with a decay rate $\alpha < 1$

**Algorithm 1** Example of FedGSNR in conjunction with FedAvg

**Input:** initial model $\boldsymbol{w}_1$, learning rate $\eta_0$, sample size $B$, and chosen global steps $M$
**for** $r = 1$ **to** $R$ **do**
  Sample clients $\boldsymbol{S} \subseteq \{1, \cdots, K\}$
  **Server**: send $\boldsymbol{w}_1$ and $\eta_r$ to each client $i \in \boldsymbol{S}$
  **On each active client** $i$ **in parallel**: initialize local model $\boldsymbol{w}^i \leftarrow \boldsymbol{w}_1$, compute $\tilde{\boldsymbol{g}}_i$ and $diag(\tilde{\boldsymbol{\Sigma}}_i)$, and send them to the server
  **Server**: compute $n_{1,i}^{opt}$ with Theorem 3.5 for each client $i$, and send it to each client $i$
  **On each active client** $i$ **in parallel**:
    **for** $k = 1$ **to** $M \cdot n_{1,i}^{opt}$ **do**
      $\boldsymbol{w}^i \leftarrow \boldsymbol{w}^i - \eta_r \bar{\boldsymbol{g}}_i$
    **end for**
  **Server**: $\boldsymbol{w}_1 \leftarrow \sum_{i=1}^{|\tilde{\boldsymbol{S}}|} \tilde{p}_i \boldsymbol{w}^i$, where $\tilde{\boldsymbol{S}} = \{i | i \in \boldsymbol{S}, n_{1,i}^{opt} > 0\}$, and $\tilde{p}_i$ is the corresponding probability ratio, i.e., $p_i / \sum_{k \in \tilde{\boldsymbol{S}}} p_k$
**end for**

To estimate the optimal local updates, we need to compute the mean vectors and covariance matrix for both local distribution and global distribution by the samples from each participant. In practice, we use sample mean vector and sample covariance matrix to estimate the parameters of local distribution, i.e., for participant $k$ with model $\boldsymbol{w}_1$, the corresponding statistics are $\tilde{\boldsymbol{g}}_k = \frac{1}{B}\sum_{b=1}^{B} \boldsymbol{g}_{k,b}$ and $\tilde{\boldsymbol{\Sigma}}_k = \frac{1}{B}\sum_{b=1}^{B}(\boldsymbol{g}_{k,b} - \tilde{\boldsymbol{g}}_k)(\boldsymbol{g}_{k,b} - \tilde{\boldsymbol{g}}_k)^T$, where $\boldsymbol{g}_{b,k} = \nabla_{\boldsymbol{w}} F(\boldsymbol{w}_1, \boldsymbol{\xi}_b | C_k)$. While for the server, based on the theorem of conditional random variables, the corresponding global statistics are

$$\tilde{\boldsymbol{g}} = \mathbb{E}_C[\tilde{\boldsymbol{g}}|C] = \sum_{k=1}^{K} p_k \tilde{\boldsymbol{g}}_k, \tag{8}$$

$$\tilde{\boldsymbol{\Sigma}} = \mathbb{E}_C[\tilde{\boldsymbol{\Sigma}}|C] + Cov_C(\tilde{\boldsymbol{g}}|C) = \sum_{k=1}^{K} p_k \tilde{\boldsymbol{\Sigma}}_k + \sum_{k=1}^{K} p_k [(\tilde{\boldsymbol{g}}_k - \tilde{\boldsymbol{g}})(\tilde{\boldsymbol{g}}_k - \tilde{\boldsymbol{g}})^T]. \tag{9}$$

In practice, as the covariance matrix increases the communication traffic and the calculation of matrix introduces lots of computation, we need to simplify the procedure. Specifically, in Theorem 3.5, we mainly need the trace of covariance matrix. Meanwhile, according to Balduzzi et al. (2017), we know that the covariance matrix of gradients is a sparse matrix and the estimate error can be scaled by the batch size $B$. Therefore, we can instead utilize the principal diagonal element of $\tilde{\boldsymbol{\Sigma}}_k$ for efficiency. Based on former analysis, we propose an algorithm FedGSNR to calculate the optimal number of local updates, and Algorithm 1 is a typical example of FedGSNR in conjunction with FedAvg[2].

**Partial participation.** In federated scenarios, the active participants are usually not 100%, so we cannot obtain perfect information of global gradient. However, we claim that FedGSNR can also adapt to this situation, because such imperfect information encourages us to transform Eq. (5) with triangle inequality to

$$\boldsymbol{W}_2(p_{\bar{\boldsymbol{w}}}(\boldsymbol{w}), p_{\boldsymbol{w}_g}(\boldsymbol{w})) \leq \boldsymbol{W}_2(p_{\bar{\boldsymbol{w}}}(\boldsymbol{w}), p_{\hat{\boldsymbol{w}}}(\boldsymbol{w})) + \boldsymbol{W}_2(p_{\hat{\boldsymbol{w}}}(\boldsymbol{w}), p_{\boldsymbol{w}_g}(\boldsymbol{w})),$$

where $\hat{\boldsymbol{w}}$ is the average parameters of active clients (i.e., a subset of total clients). With a specific client set $\boldsymbol{S}$, $\delta = \boldsymbol{W}_2(p_{\hat{\boldsymbol{w}}}(\boldsymbol{w}), p_{\boldsymbol{w}_g}(\boldsymbol{w}))$ is a constant, and $\boldsymbol{W}_2(p_{\bar{\boldsymbol{w}}}(\boldsymbol{w}), p_{\hat{\boldsymbol{w}}}(\boldsymbol{w}))$ can be bounded by inequality (6). Thus, with some tolerance $\delta$, we can similarly minimize Eq. (5) with the new upper bound, but the performance decreases as the ratio of active clients declines.

**Convergence analysis.** The convergence of FedGSNR is obvious, as we just change the number of local updates. Moreover, if we denote $E_{max} = \max\{E_i\}_{i=1}^{K}$, we can easily transform the convergence analysis of FedGSNR to its original version by the inequality $E_i \leq E_{max}$. Therefore, the convergent property of FedGSNR is resemble to its original version.

[2]Note that our proposed FedGSNR is a compatible method, and the referred FedAvg can also be replaced by other methods (e.g., FedProx).

## 4 CALCULATE GSNR BY LOCAL GRADIENTS

In this section, we first analyze the optimal local updates as well as the optimal distance between the local gradient distribution and the global gradient distribution, and then derive a method to calculate GSNR by the optimal distance. However, due to the limited space, the detailed analysis between GSNR and the optimization procedure can be found in Appendix C.

Regarding the optimal number of local updates $n_1^{opt}$ and the corresponding optimal distance $(\Delta)^2$, let $L = \|\boldsymbol{\mu}_g\|^2 + \frac{tr(\boldsymbol{\Sigma}_g)}{B}$, $M = \boldsymbol{\mu}_l^T \boldsymbol{\mu}_g + \frac{tr((\boldsymbol{\Sigma}_l \boldsymbol{\Sigma}_g)^{\frac{1}{2}})}{B}$, and $N = \|\boldsymbol{\mu}_l\|^2 + \frac{tr(\boldsymbol{\Sigma}_l)}{B}$, we can rewrite $n_1^{opt} = \max(0, \frac{M}{N})$ and $(\Delta)^2 = L - \frac{M^2}{N}$.

Then we define a matrix as follows:

$$\boldsymbol{R}_* = \left(\begin{array}{cccc:c} u_*^1 & & & & \\ & u_*^2 & & & \\ & & \ddots & & \\ & & & u_*^t & \\ \hdashline & & & & \left(\frac{1}{B}\boldsymbol{\Sigma}_*\right)^{\frac{1}{2}} \end{array}\right)$$

where $\mu_*^i$ is the component of $\boldsymbol{\mu}_* = (\mu_*^1, \cdots, \mu_*^t)$.

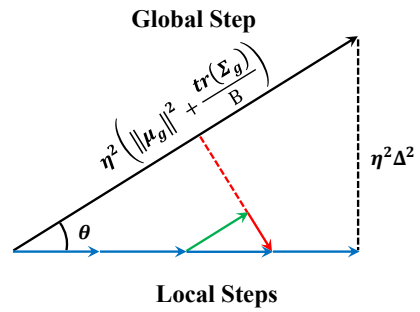



Figure 2: An overview for GSNR: the GSNR can be calculated by the statistics of global gradient distribution and local gradient distribution.

On the one hand, $L = \|\boldsymbol{R}_g\|_F^2$ is correlated to the global distribution, which is a constant for each participant.[3] On the other hand, $N = \|\boldsymbol{R}_l\|_F^2$ only depends on local distribution, thus it can be viewed as a normalization coefficient. Hence, the two variables $n_1^{opt}$ and $(\Delta)^2$ mainly depend on the value of $M$, the inner product of two matrixes, i.e., $< \boldsymbol{R}_l, \boldsymbol{R}_g >_F$, which represents the similarity of the corresponding matrixes.

Based on former analysis, we derive a method to calculate GSNR as Definition 4.1.

**Definition 4.1. Gradient Signal to Noise Ratio (GSNR).** For a local dataset $\boldsymbol{D}_l$ and a global dataset $\boldsymbol{D}_g$, with a loss function $F(\boldsymbol{w}, \cdot)$, the GSNR is a function of $\boldsymbol{w}$ defined as

$$r(\boldsymbol{w}) = \max(0, \frac{< \boldsymbol{R}_l, \boldsymbol{R}_g >_F}{\sqrt{\|\boldsymbol{R}_l\|_F^2 \|\boldsymbol{R}_g\|_F^2 - < \boldsymbol{R}_l, \boldsymbol{R}_g >_F^2}}).$$

Specifically, Fig. 2 illustrates an example for Definition 4.1. In Fig. 2, we imagine a similar case in Euclid space. In this case, angle $\theta$ can be viewed as the similarity between global and local gradient vectors. In Euclid space, the minimum distance from a point to a line is the segment vertical to the line, thus the black dash line is orthogonal to local gradient vector. Hence, $\frac{\eta^2(\|\boldsymbol{\mu}_g\|^2 + tr(\boldsymbol{\Sigma}_g)/B)}{\eta^2\Delta^2} = csc^2\theta$. As for GSNR, defined as the magnitude ratio between the parallel component and the orthogonal component, it can be viewed as the $\cot\theta$. Then according to trigonometric transformation, i.e., $\cot^2\theta = \csc^2\theta - 1$, we can obtain Definition 4.1.

## 5 EXPERIMENT

We run our experiments on the well known real world datasets CIFAR-10 and CIFAR-100 mentioned in Krizhevsky et al. (2009) to validate our design.

**Setup.** For non-iid settings, we utilize 3 methods for data partition. First, we follow the settings in Hsu et al. (2019) to generate non-iid data across different participants by Dirichlet distribution, where $\alpha$ is a parameter represents the level of non-iid. Second, we propose NonBalance and Pareto for imbalanced partition, which simulates the imbalanced distributed information in practical scenario. Due to the limited space, the details of different methods can be found in Appendix D. For all experiments, we use LeNet for CIFAR-10 and VGG-16 for CIFAR-100.

[3] $\|\cdot\|_F$ and $< \cdot, \cdot >_F$ are Frobenius inner product and Frobenius norm respectively.

Table 1: Communication rounds to reach 0.5 accuracy and corresponding speedup[5] of FedGSNR on CIFAR10. We distributed the data among 30 clients, utilize batch size of 64, and set $E_{const} = 20$.

| Algorithms | $\alpha = 0.5$ | $\alpha = 0.1$ | Label 2 | NonBalance | Pareto |
|---|---|---|---|---|---|
| FedAvg | 170 (1.0×) | 355 (1.0×) | 470 (1.0×) | 210 (1.0×) | 550 (1.0×) |
| FedGSNR with FedAvg | **115 (1.5×)** | **285 (1.3×)** | **340 (1.4×)** | **155 (1.4×)** | **170 (3.2×)** |
| FedProx | 185 (1.0×) | 430 (1.0×) | 500 (1.0×) | 220 (1.0×) | 385 (1.0×) |
| FedGSNR with FedProx | **140 (1.3×)** | **300 (1.4×)** | **405 (1.2×)** | **180 (1.2×)** | **210 (1.8×)** |
| Scaffold | 385 (1.0×) | 770 (1.0×) | 870 (1.0 ×) | 390 (1.0 ×) | >1K |
| FedGSNR with Scaffold | **140 (2.7×)** | **425 (1.8×)** | **770 (1.1×)** | **170 (2.3×)** | >1K |

To ensure all methods are comparable, we need to set the total computation, i.e., local updates, to be equal. So in FedGSNR, we set the local updates for different participants to be $E_k = NE_{const}\frac{n_{1,k}^{opt}}{\sum_{i=1}^{K} n_{1,i}^{opt}}$, where $N$ and $E_{const}$ represent the active participants and the local updates of baseline algorithms respectively. Note that $E_k$ is a redistribution of local steps.

**The necessity of optimal local updates.** To understand the necessity of optimal local updates, we calculate the entropy of the local steps, i.e., $H = \sum_{k=1}^{K} p(C_k) \log p(C_k)$, where $p(C_k) = \frac{n_{1,k}^{opt}}{\sum_{i=1}^{K} n_{1,i}^{opt}}$, and the results are illustrated in Fig. 3. Specifically, the dashed blue line on the top is the uniform distribution of local updates, which is the maximum entropy distribution of discrete variables, and it represents the equal local updates among all participants. Moreover, when the degree of non-iid increases, the computation is allocated more concentrated, i.e., a smaller entropy. On the contrary, the entropy converges to its maximum when the distributed data is closer to iid. Additionally, the corresponding convergence rate is illustrated in Table 1, which indicates that the re-allocation of local updates based on FedGSNR accelerates the model convergence.

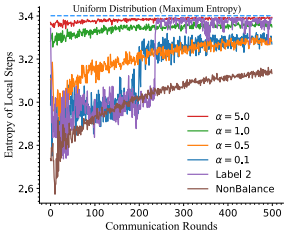



Figure 3: The entropy of local steps for different partition methods.

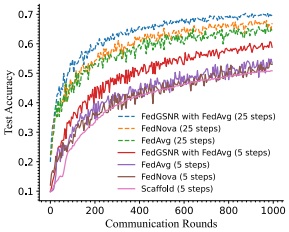
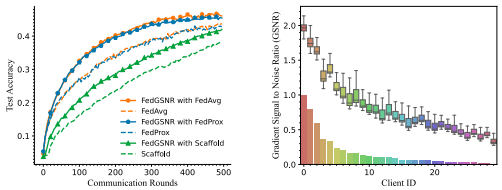


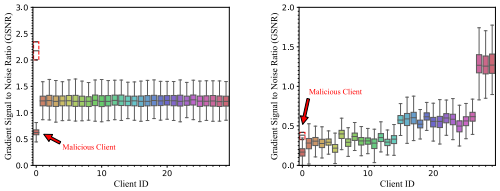

Figure 4: Test accuracy of different algorithms with different local steps.

**The impatct on test accuracy.** Versus its original, FedGSNR achieves comparable test accuracy and even outperform its original when the non-iid degree increases. For example, in Pareto scenario, the accuracy of FedGSNR with FedProx achieves an increase of 6.43%. More detailed results can be found in Appendix D.2.




(a) Test accuracy (b) GSNR

Figure 5: An imbalanced scenario with Pareto partition on CIFAR-100 dataset.




(a) IID (b) NonBalance

Figure 6: The variation of GSNR when we change the labels of a specific participant.

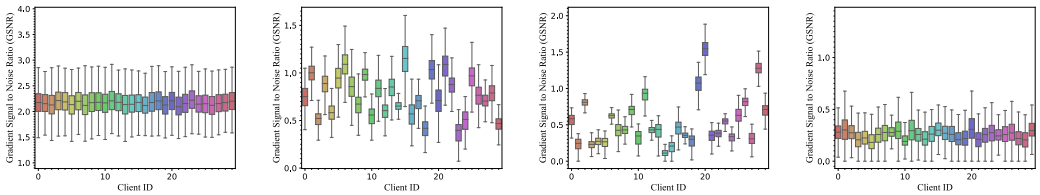
**Model convergence of FedGSNR.** During experiments, we set $\mu = 0.01$ for FedProx, and compare the performance of different algorithms to its combination with FedGSNR. Then according to Table 1, FedGSNR with different algorithms achieve faster convergence versus its original, and it reaches a $1.69\times$ speedup on average with comparable accuracy. The corresponding accuracy can be found in Appendix D.2. Besides, Fig. 4 illustrates the accuracy when we set different number of $E_{const}$, and FedGSNR with FedAvg converges faster and achieves better accuracy with different local steps (Scaffold fails to work when we set $E_{const} = 25$). Moreover, in practice, Pareto's Law is a common

[5]Speedup Karimireddy et al. (2020), i.e., $S = \frac{T_{old}}{T_{new}}$, measures the relative performance of two methods.

Table 2: Communication rounds to reach 0.5 test accuracy for classification on NonBalance CIFAR-10 of 100 participants as we vary the number of active clients.

| | 10% | 20% | 100% |
|---|---|---|---|
| FedAvg | 210 (1.0×) | 140 (1.0×) | 100 (1.0×) |
| FedNova | 235 (0.9×) | 140 (1.0×) | 80 (1.2×) |
| Scaffold | 230 (0.9×) | - | - |
| FedGSNR with FedAvg | **150 (1.4×)** | **80 (1.7×)** | **55 (1.8×)** |

principle, which means a small number of participants possess a large number of information. Fig. 5(a) indicates that FedGSNR with different algorithms converge faster and reaches comparable accuracy. Meanwhile, the GSNR of different participants are resemble to their label distribution (the histogram on the bottom of Fig. 5(b)), which demonstrates GSNR can distinguish the information quality between different local datasets. Furthermore, Table 2 indicates that the growth of active clients speeds up the convergence of different algorithms. Particularly, FedGSNR gains more benefit from global information as its speedup increases from $1.4\times$ to $1.8\times$ when active clients grows.




(a) IID (b) Dirichlet(1.0) (c) Dirichlet(0.5) (d) Label 2

Figure 7: The GSNR of different clients. We observe that GSNR is larger and almost the same among all participants when data is iid distributed, then it gets smaller and heterogeneous as the non-iid level grows. Finally, when data partition method is Label 2, GSNR is small but similar to each other again, which probably indicates that data is distributed with some symmetries in regard to the information.

**Evaluate local contributions with GSNR.** Fig. 7 displays the variation of GSNR when we utilize Dirichlet method with different $\alpha$. And the results demonstrate that when the level of non-iid grows, GSNR of different clients vary dramatically, which represents the contributions of different clients are different. Moreover, combined GSNR with the results in Table 1, the model convergence is faster than its opponents when we considerate such differences. Further more, to investigate the performance of data evaluation, we change the labels $l$ of client 0 to be $(l + k)\, mod\, 10$. So that the client provides a label flipping attack as Hitaj et al. (2017). Fig. 6 illustrates the changes of GSNR when we change the labels, and the red dashed line box represents the original GSNR when the labels are unchanged. Specifically, we observe that the GSNR dramatically decreases when we make a malicious change to the labels. Additionally, we instead change the data points to be sampled from a uniform distribution, and observe a similar phenomenon. For both of malicious changes, we observe FedGSNR is more robust. Due to the limited space, we send these experiments to Appendix D.

## 6 Conclusion

In this paper, we have investigated the FL problem via a new perspective, i.e., GSNR. Our theoretical analysis indicates that under non-iid scenarios, the local updates can be decomposed into signal and noise components, and we can maximize GSNR with the optimal local updates. Based on theoretical analysis, we further propose an algorithm FedGSNR to calculate the optimal local updates for different FL algorithms, which achieves faster global model convergence. Additionally, we derive a method to calculate GSNR directly from the local datasets, which can be utilized to evaluate the local contributions of different participants. Finally, extensive experimental results demonstrate the beneficial effect of optimizing FL from the new perspective of GSNR, and also open up a promising new direction for follow-up research.

## REFERENCES


Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N. Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *Proceedings of ICLR*. OpenReview.net, 2021.

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How To Backdoor Federated Learning. In *Proceedings of AISTATS*, pp. 2938–2948, 2018.

David Balduzzi, Marcus Frean, Lennox Leary, J. P. Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *Proceedings of ICML*, 2017.

Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *Proceedings of USENIX Security*, pp. 1605–1622, 2020.

Farzin Haddadpour and Mehrdad Mahdavi. On the Convergence of Local Descent Methods in Federated Learning. *CoRR*, abs/1910.14425, 2019.

Farzin Haddadpour, Mohammad Mahdi Kamani, Mehrdad Mahdavi, and Viveck R. Cadambe. Trading redundancy for communication: Speeding up distributed SGD for non-convex optimization. In *Proceedings of ICML*, volume 97, pp. 2545–2554. PMLR, 2019.

Briland Hitaj, Giuseppe Ateniese, and Fernando Pérez-Cruz. Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning. In *Proceedings of CCS*, pp. 603–618. ACM, 2017.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *CoRR*, abs/1909.06335, 2019.

Yihan Jiang, Jakub Konečný, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *CoRR*, abs/1909.12488, 2019.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In *Proceedings of ICML*, pp. 5132–5143, 2020.

Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. First Analysis of Local GD on Heterogeneous Data. *CoRR*, abs/1909.04715, 2019.

Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. Tighter Theory for Local SGD on Identical and Heterogeneous Data. In *Proceedings of AISTATS*, pp. 4519–4529, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. Survey of personalization techniques for federated learning. *CoRR*, abs/2003.08673, 2020.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated Optimization in Heterogeneous Networks. In *Proceedings of MLSys*, 2020a.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and Robust Federated Learning Through Personalization. In *Proceedings of ICML*, pp. 6357–6368, 2021.

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the Convergence of FedAvg on Non-IID Data. In *Proceedings of ICLR*, 2020b.

Jinlong Liu, Yunzhi Bai, Guoqing Jiang, Ting Chen, and Huayan Wang. Understanding Why Neural Networks Generalize Well Through GSNR of Parameters. In *Proceedings of ICLR*, 2020.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *CoRR*, abs/2002.10619, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of AISTATS*, pp. 1273–1282, 2017.

Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting Unintended Feature Leakage in Collaborative Learning. In *Proceedings of IEEE SP (Oakland)*, pp. 691–706, 2019.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic Federated Learning. In *Proceedings of ICML*, pp. 4615–4625, 2019.

Yurii Nesterov. Introductory lectures on convex programming volume i: Basic course. *Lecture notes*, 3(4):5, 1998.

Tom Rainforth, Adam R. Kosiorek, Tuan Anh Le, Chris J. Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. Tighter Variational Bounds are Not Necessarily Better. In *Proceedings of ICML*, pp. 4274–4282, 2018.

Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive Federated Optimization. In *Proceedings of ICLR*, 2021.

Khe Chai Sim, Petr Zadrazil, and Françoise Beaufays. An investigation into on-device personalization of end-to-end automatic speech recognition models. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pp. 774–778. ISCA, 2019.

Rachael Hwee Ling Sim, Yehong Zhang, Mun Choon Chan, and Bryan Kian Hsiang Low. Collaborative machine learning with incentive-aware model rewards. In *Proceedings of ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8927–8936. PMLR, 2020.

Cédric Villani. *Optimal transport: old and new*, volume 338. 2009.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris S. Papailiopoulos, and Yasaman Khazaeni. Federated Learning with Matched Averaging. In *Proceedings of ICLR*, 2020a.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. In *Proceedings of NeurIPS*, 2020b.

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.

Dong Yin, Ashwin Pananjady, Maximilian Lam, Dimitris S. Papailiopoulos, Kannan Ramchandran, and Peter L. Bartlett. Gradient diversity: a key ingredient for scalable distributed learning. In *Proceedings of AISTATS*, volume 84, pp. 1998–2007. PMLR, 2018.

Hao Yu, Sen Yang, and Shenghuo Zhu. Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of AAAI*, pp. 5693–5700. AAAI Press, 2019.

Ligeng Zhu, Zhijian Liu, and Song Han. Deep Leakage from Gradients. In *Proceedings of NeurIPS*, pp. 14747–14756, 2019.

# A Related Work

**Gradient Diversity.** Gradient diversity is a key ingredient of federated learning, which captures the differences between the datasets possessed by different participants. Yin et al. (2018) employs gradient diversity to investigate the relationship between batch size and the convergence rate in parallel SGD. Yu et al. (2019) analyzes why periodical model averaging is suitable for deep learning, and provides a deep understanding of model averaging. Haddadpour et al. (2019) tries to mitigate gradient diversity through sharing a small batch of data among all participants, but it also introduces a higher privacy risk. Acar et al. (2021) introduces a dynamic regularization term to resolve the problem of gradient divergence. However, most of previous literatures try to solve gradient diversity through gradient calculating, such as gradient prediction, regularization, personalized target function and so on. However, the influence of local updates gains less attention. In this paper, we propose a new perspective to analyze the optimization procedure by Gradient Signal to Noise Ratio, it reduces the required communication rounds via an elaborate configuration of local updates and propose a method to evaluate the contributions of different participants.

**Personalization in federated learning.** Another important problem in federated learning is personalization. Formally, personalization transform the optimization problem from global distribution $p(\boldsymbol{\xi})$ to a specific local distribution $p(\boldsymbol{\xi} \mid C_i)$ on client $i$, it scarifies the global performance in order to gain more benefit in local scenario. Kulkarni et al. (2020) reviews the investigations of personalization, and the situations are divided into three categories: device heterogeneity, data heterogeneity and model heterogeneity, while the last one is the motivation of personalization. Mansour et al. (2020) proposes three methods to achieve personalization, these methods try to balance the model performance on global data distribution as well as local data distribution. Sim et al. (2019) proposes a method to optimize global model and local model separately in order to make local model more personalized. Jiang et al. (2019) proposes three objectives to make personalization easily. However, personalization is an important topic in federated learning since different participants confront different problems, but if we greedily utilize global information for personalization, there is likely to appear Prisoner's Dilemma, the collective benefit for all participants is not optimal and therefore, the profit for each participant can probably be futher improved. Hence, cooperation is also an important problem, and a better goal of personalization is to search optimum on conditional data distribution $p(\boldsymbol{\xi} \mid C_i)$ combined with cooperation.

# B Proofs of Lemma and Theorem

## B.1 Proof of Lemma 3.2

*Proof.* For any constant $n$, gradient vector $\boldsymbol{g}$ can be rewritten as

$$\boldsymbol{g} = \frac{1}{B}\sum_{b=1}^{B}(\nabla_{\boldsymbol{w}} F(\boldsymbol{w}_1, \boldsymbol{\xi}_{1,b}|C_k), \cdots, \nabla_{\boldsymbol{w}} F(\boldsymbol{w}_n, \boldsymbol{\xi}_{n,b}|C_k)),$$

let $\tilde{\boldsymbol{g}}_b = (\nabla_{\boldsymbol{w}} F(\boldsymbol{w}_1, \boldsymbol{\xi}_{1,b}|C_k)), \cdots, \nabla_{\boldsymbol{w}} F(\boldsymbol{w}_n, \boldsymbol{\xi}_{n,b}|C_k)))$, we further have

$$\boldsymbol{g} = \frac{1}{B}\sum_{b=1}^{B}\tilde{\boldsymbol{g}}_b,$$

then with Assumption 3.1 and $n$ is a constant, $\tilde{\boldsymbol{g}}_b$ is subject to some complex distribution with bounded covariance matrix. As $\boldsymbol{\xi}_{i,b}|C_k$ is sampled from the same distribution independently, $\boldsymbol{g}$ is the mean vector of $\tilde{\boldsymbol{g}}_1, \cdots, \tilde{\boldsymbol{g}}_B$, which are iid random vectors.
Therefore, due to the classical Central Limit Theory, with $B$ growing large, $\sqrt{B}(\boldsymbol{g} - \mathbb{E}[\boldsymbol{g}])$ converges to $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ in distribution, where $\boldsymbol{\Sigma}$ is the covariance matrix of $\boldsymbol{g}$. □

## B.2 Proof of Lemma 3.4

*Proof.* First, we prove $\lim_{r\to+\infty} \mathbb{E}\|\boldsymbol{\varepsilon}_r\| = 0$.

Regarding the gradient $\boldsymbol{g}_i$, $i \in \{1, \cdots, n\}$, due to the smoothness of $\nabla_{\boldsymbol{w}} F(\boldsymbol{w}_i, \boldsymbol{\xi})$, we can expand $\boldsymbol{g}_i$ based on Lagrange's mean value theorem as

$$\boldsymbol{g}_i = \nabla_{\boldsymbol{w}} F(\boldsymbol{w}_i, \boldsymbol{\xi}_i) = \nabla_{\boldsymbol{w}} F(\boldsymbol{w}_1, \boldsymbol{\xi}_i) + H(F(\tilde{\boldsymbol{w}}_i,\, \boldsymbol{\xi}_i))(\boldsymbol{w}_i - \boldsymbol{w}_1) = \boldsymbol{g}_{1,i} + H(F(\tilde{\boldsymbol{w}}_i,\, \boldsymbol{\xi}_i))(\boldsymbol{w}_i - \boldsymbol{w}_1), \tag{10}$$

where $\tilde{\boldsymbol{w}} := \lambda \boldsymbol{w}_i + (1-\lambda)\boldsymbol{w}_1$, $\lambda \in [0, 1]$, and $\boldsymbol{g}_{1,i}$ represents the gradient at $\boldsymbol{w}_1$ with sample $\boldsymbol{\xi}_i$, which is an unbiased estimator of $\boldsymbol{g}_1$. Note that when $i \neq j$, $\boldsymbol{g}_{1,i}$ is independent of $\boldsymbol{g}_{1,j}$. Then

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{g}_i - \boldsymbol{g}_{1,i}\| &= \mathbb{E}\|H(F(\tilde{\boldsymbol{w}}_i,\, \boldsymbol{\xi}_i))(\boldsymbol{w}_i - \boldsymbol{w}_1)\| \overset{(a)}{\leq} \mathbb{E}\|H(F(\tilde{\boldsymbol{w}}_i,\, \boldsymbol{\xi}_i))\|\|(\boldsymbol{w}_i - \boldsymbol{w}_1)\| \\
&\overset{(b)}{\leq} \sqrt{\mathbb{E}\|H(F(\tilde{\boldsymbol{w}}_i,\, \boldsymbol{\xi}_i))\|^2 \mathbb{E}\|\boldsymbol{w}_i - \boldsymbol{w}_1\|^2} \overset{(c)}{\leq} L \cdot \sqrt{\mathbb{E}\|\eta_r \sum_{j=1}^{i-1} \boldsymbol{g}_j\|^2} \\
&\overset{(d)}{\leq} L^2 \cdot \eta_r \sqrt{(i-1)\sum_{j=1}^{i-1} \mathbb{E}\|\boldsymbol{g}_j\|^2} \overset{(e)}{\leq} (i-1)\eta_r L G,
\end{aligned} \tag{11}$$

where $(a)$ follows from sub-multiplicative property of matrix norm, $(b)$ is based on Cauchy–Schwarz inequality, $(c)$ is an immediate consequence of Assumption 3.3 and the local optimization process, $(d)$ comes from the fact $\|\sum_{i=1}^n \boldsymbol{a}_i\|^2 \leq n \sum_{i=1}^n \|\boldsymbol{a}_i\|^2$, and (e) is based on Assumption 3.1, where $G^2 := \sigma^2 + \mu^2$, $\mu = \max(\{\|\boldsymbol{\mu}_i\|\}_{i \in \{1, \cdots, n\}})$. Hence,

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{\varepsilon}_r\| = \mathbb{E}\|\hat{\boldsymbol{g}} - \bar{\boldsymbol{g}}\| &= \mathbb{E}\|\sum_{i=1}^n (\boldsymbol{g}_i - \boldsymbol{g}_{1,i})\| \leq \sum_{1}^n \mathbb{E}\|\boldsymbol{g}_i - \boldsymbol{g}_{1,i}\| \\
&\leq (\sum_{i=1}^n (i-1))\eta_r L G = \frac{n(n-1)}{2}\eta_r L G
\end{aligned}$$

where the first inequality follows from the triangle inequality, and the second inequality is based on Eq. (11).

As $n$ represents the number of local steps, which is a constant, $\mathbb{E}\|\boldsymbol{\varepsilon}_r\|$ is upper bounded by $\eta_r \cdot M$, where $M$ is a bounded value. Therefore,

$$0 \leq \lim_{r \to +\infty} \mathbb{E}\|\boldsymbol{\varepsilon}_r\| \leq \lim_{r \to +\infty} \eta_r \cdot M = 0. \tag{12}$$

Formula (12) implies $\boldsymbol{\varepsilon}_r$ converge to 0 in mean, i.e., $\boldsymbol{\varepsilon}_r \xrightarrow{L} 0$, which immediately completes the proof.

□

### B.3 Proof of Theorem 3.5

*Proof.* Implied by Lemma 3.2, one global update is a specific case, and with a finite batch size $B$, the global gradient distribution can be approximately interpreted as $\mathcal{N}(\eta_r \boldsymbol{\mu}_g, \frac{\eta_r^2 \boldsymbol{\Sigma}_g}{B})$. While for $n$ local updates, based on Lemma 3.2 and Lemma 3.4, it approximately conforms to $\mathcal{N}(n\eta\boldsymbol{\mu}_l, n^2\eta^2 \frac{\boldsymbol{\Sigma}_l}{B})$. Then according to Eq. (4), to minimize the distance between global and local gradients, we can build an optimization problem as

$$\begin{aligned}
\min_{n} \quad & d^2 = \|\eta\boldsymbol{\mu}_g - n\eta\boldsymbol{\mu}_l\|^2 + tr(\boldsymbol{M}^2) \\
\text{s.t.} \quad & \boldsymbol{M} = \left(\frac{\eta^2 \boldsymbol{\Sigma}_g}{B}\right)^{\frac{1}{2}} - \left(\frac{n^2\eta^2 \boldsymbol{\Sigma}_l}{B}\right)^{\frac{1}{2}} \\
& n \geq 0.
\end{aligned} \tag{13}$$

Note that Eq. (13) is a quadratic function of $n$, which immediately completes the proof. □

### B.4 PROOF OF COROLLARY 3.6

*Proof.* In this case, we slightly change the global gradients distribution to $\mathcal{N}(k\boldsymbol{\mu}_g, \frac{k^2\boldsymbol{\Sigma}_g}{B})$, and reformulate problem (13) as

$$\begin{aligned} \min_{n} \quad & d^2 = k^2(\|\eta\boldsymbol{\mu}_g - \frac{n}{k}\eta\boldsymbol{\mu}_l\|^2 + tr(\boldsymbol{M}^2)) \\ \text{s.t.} \quad & \boldsymbol{M} = \left(\frac{\eta^2\boldsymbol{\Sigma}_g}{B}\right)^{\frac{1}{2}} - \left(\frac{\left(\frac{n}{k}\right)^2\eta^2\boldsymbol{\Sigma}_l}{B}\right)^{\frac{1}{2}} \\ & n \geq 0, \end{aligned}$$

let $\tilde{d} = \frac{d}{k}$ and $\tilde{n} = \frac{n}{k}$, then the new problem is reduce to problem (13), which concludes the proof immediately. □

## C DETAIL ANALYSIS OF GSNR

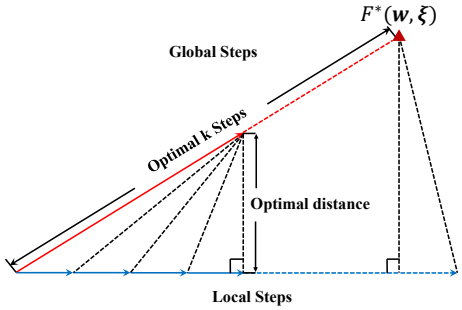



Figure 8: A similar case in Euclid space: as the ideal optimization path is discrete, there is an optimal distance between global updates and multiple local updates. Further more, when global updates converge to the optimum, the local updates converge to the nearest point to the optimum.

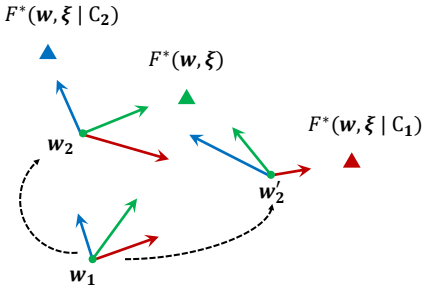



Figure 9: A representative scenario of GSNR: $r(\boldsymbol{w})$ is a random variable with regard to $\boldsymbol{w}$. If we get closer to the optimum of $C_2$, GSNR of $C_1$ will increase and the GSNR of $C_2$ will decrease. While if we get closer to the optimum of $C_1$, the situation changes in the opposite way.

According to Fig. 8, the stochastic gradient descent algorithm converges to the $\epsilon$-neighborhood of optimum after a constant steps (usually more than $\mathcal{O}(\frac{1}{\epsilon})$ Nesterov (1998)). For convenience, we refer such a constant as the optimal $m_{opt}$. In practice, as Fig. 8 indicates, with determined differences between global distribution and local distribution, i.e., maximal GSNR is a constant during a specific round, set the target global steps as $m_{opt}$ is an optimal choice. However, a large number of $m$ leads to a large error of gradient estimation, which is determined by $O(\eta^2 n^2)$. Hence, in practice, in order to decide $m$, we need to trade off between the estimation error and the corresponding convergence rate.

Then we focus on Definition 4.1, the function to calculate GSNR. Specifically, Cauchy-Schwarz inequality implies that $\Delta^2 \geq 0$, and the equality holds when local distribution is the same as the global distribution, i.e., the data is iid distributed among all participants. With $\Delta^2$ decreases, which implies local data distribution approaches the global data distribution, $n_1^{opt}$ gradually increases. Moreover, when $\Delta^2$ reaches its minimum 0, $n_1^{opt}$ attains its maximum value 1. Based on the analysis, we can conclude that the more similarity between local dataset and the global dataset, i.e., the larger GSNR the local dataset achieves, the more local updates we need for optimization procedure, which is heuristically experimented in Li et al. (2020b).

To calculate GSNR, we derive $r(\boldsymbol{w})$ as Definition 4.1, and $r(\boldsymbol{w})$ is positive related to $n_1^{opt}$. On the one hand, when $n_1^{opt} = 0$, from its definition, we know that $< \boldsymbol{R}_l, \boldsymbol{R}_g >_F = 0$, hence the optimal distance $\Delta^2$ achieves its maximum $\|\boldsymbol{R}_g\|_F^2$, and $r(\boldsymbol{w})$ attains its minimum 0. On the other hand,

when local distribution is the same as the global distribution, i.e., $\Delta^2 = 0$, $r(\boldsymbol{w}) \to +\infty$, we treat this scenario as a noiseless optimization procedure, and the data is iid distributed among each participant. Therefore, $r(\boldsymbol{w}) \in (0, +\infty)$.

With former analysis, we know $r(\boldsymbol{w}) \in (0, +\infty)$. On the one hand, as the data is iid distributed among all participants, i.e., GSNR goes to $+\infty$, the distributed optimization is a noiseless procedure, which means the local updates is unbiased. On the other hand, when GSNR is 0, which means $< \boldsymbol{R}_l, \boldsymbol{R}_g >_F \leq 0$, the angle between local gradient and the global gradient is greater than $90°$. In other words, for current optimization, local data distribution is independent of global data distribution, thus for global optimization, it is no better than a random guess, then its signal component will be set to 0, which leads GSNR to be 0.

As for the relationship between GSNR and the parameters $\boldsymbol{w}$, Fig. 9 displays a representative scenario. Due to the randomness of SGD, the new parameters after $\boldsymbol{w}_1$ with another aggregation can be either $\boldsymbol{w}_2$ or $\boldsymbol{w}_2^{'}$. On the one hand, if the parameters is $\boldsymbol{w}_2$, which means we get closer to the optimum of client $C_2$, there are different changes of the GSNR for different clients: for $C_1$, the GSNR increases, while for $C_2$, the vector is almost orthogonal to global optimization vector, which implies its GSNR is closer to 0. On the other hand, if the parameters is $\boldsymbol{w}_2^{'}$, which is closer to optimum of $C_1$, the phenomenon is slightly different: the GSNR of $C_1$ decreases and the GSNR of $C_2$ increases. Hence, during the training process, $r(\boldsymbol{w})$ is a random variable correlated to the random process $\boldsymbol{w}$, and if we tend to use GSNR to evaluate the contribution of different participants, we need to observe its statistics, i.e., mean or median.

# D Details of Experiments

## D.1 Different Methods of data partition

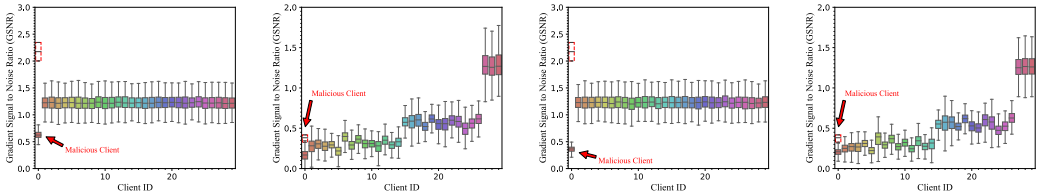
**Dirichlet partition.** we follow the settings in Hsu et al. (2019) to generate non-iid data across different participants by Dirichlet distribution. Specifically, the prior distribution is set to be Uniform, and then the parameter $\alpha$ represents the level of concentration. With $\alpha \to +\infty$, the data distributions of all participants tend to be identical, hence the data is iid distributed among all clients. While $\alpha \to 0$, each participant only possesses data chosen from just one class, i.e., one label for each participant. As for **Label 2.**, it is a specific partition method in Hsu et al. (2019), and each client owns the data sampled from 2 classes.

**NonBalance partition.** For NonBalance partition, we tend to simulate the practical scenario of imbalanced information distribution. Specifically, We divide all participants into three categories: abundant information, medium information and less information, which represent the clients possess data chosen from different number of labels. First, for clients with abundant information, we random chose data from all labels, and the number of them is $10\%$ of total clients. Second, for the clients with medium information, we random chose $50\%$ classes for each client, then distribute data randomly according to their chosen labels, and the ratio of them is $40\%$. Finally, for the clients with less information, the number of labels reduces to $20\%$, and the ratio of them raises to $50\%$.

**Pareto partition.** In practice, Pareto distribution is a common scenario. It represents the long tailed distribution of practical scenario such as the degree of nodes in complex network, the distribution of social wealth, the distribution of followers in social network, etc. Hence, we design Pareto partition to simulate the so called Two-Eight distribution in practice. First, we sample $N$ points from Pareto distribution,

$$p(x) = \begin{cases} \frac{k \cdot x_{min}^k}{x^{k+1}}, & \text{if } x \geq x_{min} \\ 0, & \text{otherwise.} \end{cases}$$

Where $N$ represents the number of clients. Denote the corresponding samples as $\boldsymbol{X} = \{x_i\}_{i=1}^N$, and we normalize $x_i$ with $\tilde{x}_i = x_i / \max(\boldsymbol{X})$ to guarantee all data distributed in [0, 1]. Finally, we use $\tilde{x}_i$ as the ratio of classes possessed by different clients for random sampling, and set the minimum number of labels among all clients to be 1.

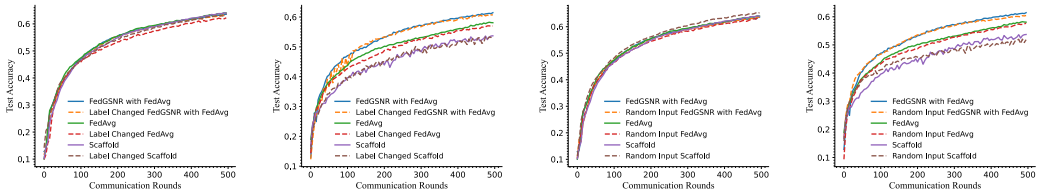



(a) GSNR on IID CIFAR-10 with Label change.

(b) GSNR on NonBalance CIFAR-10 with Label change.

(c) GSNR on IID CIFAR-10 with Random input.

(d) GSNR on NonBalance CIFAR-10 with Random input.




(e) Test accuracy on IID CIFAR-10 with Label change.

(f) Test accuracy on NonBalance CIFAR-10 with Label change.

(g) Test accuracy on IID CIFAR-10 with Random input.

(h) Test accuracy on NonBalance CIFAR-10 with Random input.

Figure 10: We make different malicious changes to client 0 in different partition methods. In (a) and (b), we change the labels $l$ of client 0 to $l+3$. While in (c) and (d), the labels remain unchanged, and we change the input data to be a uniform distribution $U(-1,1)$. (e)-(h) are the corresponding test accuracy of different scenarios.

### D.2 Additional Experiments

Fig. 10 displays the change of GSNR and corresponding test accuracy when we apply different malicious change to a client. Interestingly, compare Fig. 10(a) and 10(b) with Fig. 10(c) and 10(d), we can discover that the decrease of GSNR for random input is larger than label changing. Specifically, it is consistent with our intuition, since malicious attack of label changing contains more information than random input. Simply put, after label changing, the model still gets the information that such data belong to a same class. For example, if we change all labels of 'cat' to 'dog', we still know the data of 'cat' belongs to a same class, though we call them 'dog'. On the contrary, the change of random input cannot provide this information.

Table 3: Best test accuracy on CIFAR-10. We distributed the data among 30 clients, utilize batch size of 64, and set local steps $E_{const} = 20$ for different algorithms.

| Algorithms | $\alpha = 0.5$ | $\alpha = 0.1$ | Label 2 | NonBalance | Pareto |
|---|---|---|---|---|---|
| FedAvg | **67.06%** | 57.44% | **58.25%** | 65.48% | 57.23% |
| FedGSNR with FedAvg | 66.23% | **61.41%** | 57.69% | **66.42%** | **63.38%** |
| FedProx | 63.14% | 57.80% | **58.84%** | 63.87% | 56.96% |
| FedGSNR with FedProx | **63.5%** | **59.39%** | 58.39% | **68.17%** | **63.39%** |
| Scaffold | 62.05% | 53.98% | 50.38% | 62.79% | **40.73%** |
| FedGSNR with Scaffold | **63.53%** | **58.95%** | **55.48%** | **64.96%** | 39.83% |

Table 3 displays the corresponding test accuracy of different algorithms aforementioned in Table 1, and it indicates that FedGSNR not only converges faster, but also achieves a comparable accuracy than its opponents. As for the accuracy drop in Table 3, it's possibly because FedGSNR is a gradient-based algorithm, if the basic method introduces gradient estimation (i.e., Scaffold), the performance of FedGSNR will be correlated to the precision of such an estimation, and a relatively low precision leads to the corresponding accuracy drop. Meanwhile, as we have illustrated in Fig. 7, the method of Label 2 distributes the data with some symmetries in regard to the information, i.e., the GSNR of each participant are similar to each other, then it's naturally compatible to identical local updates, hence the test accuracy between FedGSNR and its opponents are close to each other.