FedMDFG: Federated Learning with Multi-Gradient Descent and Fair Guidance

Zibin Pan[1,2], Shuyi Wang[1,2], Chi Li[1], Haijin Wang[1], Xiaoying Tang*[1,2,3], Junhua Zhao*[1,2]

[1] The School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China
[2] The Shenzhen Institute of Artificial Intelligence and Robotics for Society
[3] The Guangdong Provincial Key Laboratory of Future Networks of Intelligence
zibinpan@link.cuhk.edu.cn, shuyiwang@link.cuhk.edu.cn, chili@link.cuhk.edu.cn, haijinwang@link.cuhk.edu.cn, tangxiaoying@cuhk.edu.cn, zhaojunhua@cuhk.edu.cn

Abstract

Fairness has been considered as a critical problem in federated learning (FL). In this work, we analyze two direct causes of unfairness in FL - an unfair direction and an improper step size when updating the model. To solve these issues, we introduce an effective way to measure fairness of the model through the cosine similarity, and then propose a federated multiple gradient descent algorithm with fair guidance (FedMDFG) to drive the model fairer. We first convert FL into a multi-objective optimization problem (MOP) and design an advanced multiple gradient descent algorithm to calculate a fair descent direction by adding a fair-driven objective to MOP. A low-communication-cost line search strategy is then designed to find a better step size for the model update. We further show the theoretical analysis on how it can enhance fairness and guarantee the convergence. Finally, extensive experiments in several FL scenarios verify that FedMDFG is robust and outperforms the SOTA FL algorithms in convergence and fairness. The source code is available at https://github.com/zibinpan/FedMDFG.

1 Introduction

Federated learning (FL) has emerged as a significant machine learning methodology where clients can utilize their local data to train a global model collaboratively without sharing data (McMahan et al. 2017). Since the global model includes significantly more local data for training, it's anticipated to have a strong generalization and perform as well as possible in all clients. Unfortunately, it's a huge challenge to train a global model that treats every client fairly (i.e., increasing the uniformity of performance across clients (Li et al. 2021b)) for many reasons, such as the data is frequently heterogeneous on clients both in terms of distribution and size, the partial selection at each communication round (Cho et al. 2020), and client dropping out (Abay et al. 2020).

Many prior techniques have been proposed to improve fairness in FL. For instance, q-FFL (Li et al. 2020c) employs a fair resource allocation mechanism; AFL (Mohri, Sivek, and Suresh 2019) tries to protect the client with the worst performance, but it only works when there are few clients (Wang et al. 2021). FedFV (Wang et al. 2021) aims to mitigate the local gradient conflicts among clients by gradient projection, but it cannot ensure to prevent of local performance reduction in the case of more than two clients; FedMGDA+ (Hu et al. 2022) designs a constraint multiple gradient descent algorithm to find a common descent direction for all clients. However, it's difficult to tune the hyperparameter of the method to achieve its goal.

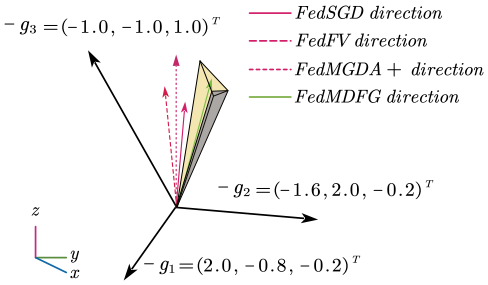

Figure 1: Directions obtained by FedMDFG and three previous gradient-based methods. g_1, g_2, and g_3 are local gradients. All possible common descent directions lie in the gray and yellow areas. The yellow area depicts all possible fair descent directions. The directions out of the gray area will cause a performance reduction for some clients.

In contrast to previous studies, we observe that the use of an unfair direction and an incorrect step size when updating the model are two direct causes of fairness violation. First and foremost, for convex problems, if the FL method finds a direction d that conflicts with some clients' gradients g_i, i.e., $\exists i, -dg_i < 0$ (Wang et al. 2021), the model accuracy will suffer reductions for these clients whatever any step size is taken. Thus, the model will become more unfair. Furthermore, as shown in Fig. 2 (b), Even if the direction is fair, a step size that is too large can undermine fairness.

To address the aforementioned issues, we propose a Federated Multi-Gradient Descent with Fair Guidance (FedMDFG) algorithm to find a fair descent direction as well as a proper step size for training the global FL model. To the best of our knowledge, FedMDFG is the first one that can determine a fair descent direction, which can not only lead to increase model performance on each client but also drive the model fairer. Fig. 1 shows that FedMDFG can discover a fair descent direction compared with some previous

*Xiaoying Tang and Junhua Zhao are corresponding authors.

algorithms, including the traditional and the SOTA methods.

We summarize our contributions as follows:

1. We introduce a simple and yet effective way to assess fairness by introducing a fair guidance vector and utilizing cosine similarity. The model with a smaller angle between the performance vector and the fair guidance vector ($p = \vec{1}$) is considered fairer.
2. We propose a novel multiple gradient descent method to establish a fair descent direction for the model update. We theoretically prove that the obtained direction can decrease each client's objective and drive the model fairer.
3. We design a low-communication-cost step size line search approach that is able to find a proper step size (learning rate) that can help ensure the model to be fairer.
4. We implement extensive experiments on several federated learning scenarios to validate that FedMDFG is robust and outperforms the SOTA algorithms in terms of fairness and convergence.

2 Background & Related Work

2.1 Problem Setup of Federated Learning

Proposed by (McMahan et al. 2017), FL allows clients to collaboratively learn models while keeping their data local (Li et al. 2020b; Kairouz et al. 2021). It can be considered as the following optimization problem that minimizes the aggregation ($\mathbf{G}$) of the local objective of clients.

$$\min_{\omega} \ \mathbf{G}(L_1(\omega), L_2(\omega), \cdots, L_m(\omega)), \tag{1}$$

where $\omega \in \mathbb{R}^n$ represents model parameters, m is the number of clients, and L_i denotes the local objective of i^{th} client, which is usually defined by the empirical risks over local data (Li et al. 2020c), i.e., $L_i(\omega^t) = \sum_{j=1}^{N_i} \frac{1}{N_i} L_{i_j}(\omega^t)$, where N_i is the number of data samples on client i. In practice, L_{i_j} is obtained by a specific loss function.

Besides, FL can be inherently treated as a multi-objective optimization problem (MOP) (Hu et al. 2022):

$$\min_{\omega} \ (L_1(\omega), L_2(\omega), \cdots, L_m(\omega)). \tag{2}$$

Several gradient-based methods have been proposed to solve Problem (2) (Désidéri 2012; Fliege and Svaiter 2000; Fliege and Vaz 2016), aiming to seek a solution that is Pareto critical (also named Pareto stationary).

Definition (Pareto-stationarity (Gebken, Peitz, and Dellnitz 2019)): ω^* is called Pareto critical iff there exists a convex combination of the gradients $\nabla L_i(\omega^*)$ that is equal to $\vec{0}$. i.e., $\sum_i^m \xi_i \nabla L_i(\omega^*) = \vec{0}$, $\xi_i \geq 0, \forall i = 1, 2, \cdots, m$, which is equivalent to $\nexists d \in \mathbb{R}^n, \nabla L(\omega)^T d < \vec{0}$.

2.2 Fairness in Federated Learning

The model's performance will likely differ significantly among clients due to the variability in the distribution and the amount of local data. (Li et al. 2021b) summaries a definition of fairness: A model ω_1 is fairer than ω_2 if the standard deviation of $L(\omega_1)$ is smaller than that of ω_2.

To increase fairness, (1) some previous works consider mini-max methods (Mohri, Sivek, and Suresh 2019; Deng, Kamani, and Mahdavi 2020). (2) Some other studies utilize reweighting approaches, for example, (Li et al. 2020c, 2021a; Zhao and Joshi 2022) propose flexible sample reweighting methods; (Huang et al. 2022) reweight clients based on their accuracy and the online time; (Li et al. 2021a) reweight by considering flexible tradeoffs between accuracy and fairness; (Kanaparthy et al. 2022) propose several aggregation functions to reweight. (3) Moreover, (Lyu et al. 2019, 2020) allocate different models to clients according to their contributions; (Li et al. 2021b) propose a personalized FL algorithm to improve fairness. (4) Recently, some techniques are proposed to find a better direction to mitigate conflicts during the model training (Wang et al. 2021; Hu et al. 2022). In contrast, we identify two direct causes of unfairness: a wrong direction and an improper step size when updating the global model. We design a novel method to find a fair descent direction and propose a low-communication-cost line search approach for determining a proper step size.

3 FedMDFG: Proposed Algorithm

The proposed algorithm, FedMDFG, (as shown in Algorithm 1), mainly includes three parts: fairness measurement (line 13), fair descent direction calculation (from line 11 to line 21), and step size line search (line 24).

In one communication round, the server initially sends the global model parameters ω^t to the new-come clients, i.e., for all client i satisfying $i \notin S_{old}, i \in S_t$. Then, for all client $i \in S_t$, they calculate the gradient g_i^t and loss L_i^t, and then upload them to the server. The server subsequently calculates a fair descent direction d^t and sends it to clients. Later, the server and clients transfer scalars to conduct the step size line search to obtain η^t. Both the server and clients utilize $\omega^{t+1} = \omega^t + \eta^t d^t$ to update their model (the model in the server and clients are the same) in the end. Algorithm 1, line 20 shows the stopping criterion of FedMDFG: it terminates when the model is Pareto critical. For convenience, the gradient of client i at round t is written as g_i^t in pseudocode, which is obtained by $g_i^t = \nabla L_i(\omega^t) = \sum_j^{N_i} \frac{1}{N_i} \nabla L_{i_j}(\omega^t)$.

3.1 Fairness Measurement

We present a simple yet effective way for assessing fairness. Since the data scale might affect the standard deviation (SD), we measure fairness via the angle (radian) between the performance vector and a fair guidance vector $p = (1, \cdots, 1)$.

Definition (Fairness): Denote a fairness indicator $\varphi(a, b) = \arccos(\frac{a \cdot b}{||a|| ||b||})$ as the inverse cosine value of the cosine similarity between vectors a and b, i.e., the angle between a and b. Given a fairness guidance vector $p = \vec{1}$, we say that model ω_1 is fairer than ω_2 if $\varphi(M(\omega_1), p) < \varphi(M(\omega_2), p)$, where the i^{th} element of the vector $M(\omega)$ reflects the performance (e.g., accuracy or local objective) on client i.

Thus, by setting a tolerable-fair angle $\theta \in (0, \frac{\pi}{2}]$, we call the model ω tolerable-fair if $\varphi(M(\omega), p) \leq \theta$, otherwise it's unfair. For example, in Fig. 2 (a), $L(\omega^1)$ is tolerable-fair and

Algorithm 1: FedMDFG

Input: Communication round T; Number of clients m; Step size line search parameter s; Tolerable fair angle θ; Learning rate η; Client online probability C.

Initialize ω^0, client record $S_{old} \leftarrow [\,]$, gradient record $g_{old} \leftarrow [\,]$, reference objectives $L_i^R \in \mathbb{R}^m$, $L_i^R = \inf, \forall i$.
for $t = 0, 1, \cdots, T-1$ **do**
  Server randomly selects a subset S_t from m clients with the prob. C, and sends the model ω^t to those client i that $i \notin S_{old}$ and $i \in S_t$.
  Server receives gradient g_i^t and the loss L_i^t from all selected clients $i \in S_t$.
  Drop i from S_t if $L_i = 0$ or $||g_i^t|| = 0$.
  $g_i^t \leftarrow g_i^t / \|g_i^t\| \cdot \text{mean}(\|g_1^t\|, \cdots, \|g_{|S_t|}^t\|), \forall i \in S_t$.
  $L(\omega^t) \leftarrow (L_1^t, \cdots, L_{|S_t|}^t)^T \in \mathbb{R}^{|S_t|}$.
  $g \leftarrow \text{concat}(g_1^t, \cdots, g_i^t, \cdots, g_{|S_t|}^t), i \in S_t$.
  Set a flag: $f \leftarrow True$.
  /* Calculate fair descent direction. */
  $Q \leftarrow g$.
  Form a guidance vector $p = \vec{1}$, $p \in \mathbb{R}^{|S_t|}$.
  if $\arccos(\frac{L(\omega^t) \cdot p}{||L(\omega^t)|| \cdot ||p||}) > \theta$ or $L_i^t > L_i^R, \exists i \in S_t$ **then**
    $h^t \leftarrow \frac{p^T L(\omega^t)}{\|L(\omega^t)\|^2} L(\omega^t) - p$, and $h^t \leftarrow h^t / \|h^t\|$.
    $Q \leftarrow \text{concat}(g, gh^t)$.
    $f \leftarrow False$.
  end if
  $Q \leftarrow \text{concat}(Q, g_{old}[S_{drop}])$, where $S_{drop} \leftarrow S_{old} \backslash S_t$.
  $d^t \leftarrow -Q\lambda$, where λ is the solution of Problem (6).
  Stop the algorithm if $d^t = \vec{0}$.
  $d^t \leftarrow \sigma d^t$, where $\sigma \leftarrow \|\frac{1}{|S_t|} \sum_i^{|S_t|} g_i^t\| / \|d^t\|$.
  The server sends d^t to all client $i \in S_t$.
  /* Step size line search (see Algorithm 2) */
  $\eta^t \leftarrow$ **LineSearch**$(\sigma, S_t, S_{old}, s, \eta, p, L(\omega^t), g^t, d^t, f)$.
  Do $\omega^{t+1} \leftarrow \omega^t + \eta^t d^t$ both in server and clients.
  Update L_i^R. (According to Section 3.2)
  $S_{old} \leftarrow S_t$ and $g_{old} \leftarrow g^t$.
end for

$L(\omega^2)$ is not. If the model is deemed unfair, we will apply the fair-enhanced strategy mentioned in Section 3.2.

3.2 Fair Descent Direction

We follow the idea of considering FL as a multi-objective optimization. (Fliege and Svaiter 2000) propose Multiple Gradient Descent Algorithm (MGDA) to solve Problem (2) by iterating $\omega^{t+1} = \omega^t + \eta^t d^t$ to render ω^t Pareto critical, where d^t is a common descent direction (i.e., $\nabla L_i(\omega)^T d < 0, \forall i$) obtained by solving Problem (3). η^t is the step size.

$$\begin{aligned} &(d^t, \alpha^t) = \arg \min_{d^t \in \mathbb{R}^n, \alpha^t \in \mathbb{R}} \alpha^t + \tfrac{1}{2}\|d^t\|^2, \\ &s.t. \quad \nabla L_i(\omega^t)^T d^t \leq \alpha^t, i = 1, \cdots, m. \end{aligned} \tag{3}$$

The solutions of Problem (3) will satisfy:

Lemma 1 (Fliege and Svaiter 2000): Let (d^t, α^t) be the solution of Problem (3).

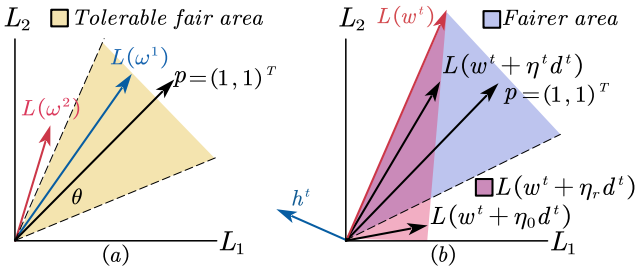

Figure 2: Tolerable fair area (a) and fairer area (b). p is the fairness guidance vector. In (b), ω^t is the model parameters. d^t is a fair descent direction. Local objective vectors of all possible updated models, i.e., $\omega^t + \eta_r d^t, \forall \eta_r > 0$, lie in the pink area. $\omega^t + \eta_0 d^t$ is a more unfair model updated by η_0 that is too large. η^t is a proper step size to enhance fairness.

1. If ω^t is Pareto critical, then $d^t = \vec{0}$ and $\alpha^t = 0$.
2. If ω^t is not Pareto critical, then

$$\begin{aligned} &\alpha^t \leq -(1/2)\|d^t\|^2 < 0, \\ &\nabla L_i(\omega^t)^T d^t \leq \alpha^t, i = 1, \cdots, m, \end{aligned} \tag{4}$$

where d^t is a common descent direction that can decrease each client's objective when the model isn't Pareto critical.

This method has been utilized in FedMGDA (Hu et al. 2022). However, it cannot prevent the variance of reductions of clients' objectives, which may harm fairness. Fig. 2 (a) illustrated a case where all objectives of ω^2 are lower than ω^1 while the first objective shrinks significantly relative to the second, resulting in a more unfair model. Even worse, the model is prone to be local-overfitting, i.e., $\exists i, j, i \neq j, L_i(\omega^t) = 0$ but $L_j(\omega^t) > 0$. We follow the idea of MGDA to calculate a common descent direction when the model is tolerable-fair and propose a fair-enhanced strategy to derive a fair descent direction when the model is judged unfair.

Enhance Fairness. We add an extra fair-driven objective: $\min_\omega L(\omega)^T h^t$ to Problem (2) to direct the model toward fairness. Specifically, we establish a temporary Problem (5) at round t, where $h^t = \text{normalize}(\frac{p^T L(\omega^t)}{\|L(\omega^t)\|^2} L(\omega^t) - p)$ is the opposite normalized vector of the projection of p on the normal plane of $L(\omega^t)$, which can be seen in Fig. 2 (b).

$$\min_\omega \; (L_1(\omega), L_2(\omega), \cdots, L_m(\omega), L(\omega)^T h^t). \tag{5}$$

We can obtain a common descent direction d^t for Problem (5) by adding a constraint: $(\nabla L(\omega^t) h^t)^T d^t \leq \alpha^t$ to Problem (3) and solve it. According to Theorem 1, where the proof can be seen in Appendix A.1, the Pareto critical solution of Problem (5) must also be Pareto critical in Problem (2). So there exists $\eta_0 > 0$ such that for $\omega^{t+1} = \omega^t + \eta_0 d^t$, it satisfies $L_i(\omega^{t+1}) < L_i(\omega^t), \forall i$ and $L(\omega^{t+1})^T h^t < L(\omega^t)^T h^t = 0$. Furthermore, we can obtain an η^t, that $\eta_0 \geq \eta^t > 0$ and $\varphi(L(\omega^t + \eta^t d^t), p) < \varphi(L(\omega^t), p)$ to enhance fairness (Fig. 2 (b)). In Section 3.3, we will describe how to get a proper η^t.

Theorem 1. For a non-Pareto critical solution ω^t of Problem (2), there always exists a direction d^t, such that $\nabla L_i(\omega^t)^T d^t < 0, i = 1, \cdots, m$ and $(\nabla L(\omega^t) h^t)^T d^t < 0$.

Dimension Reduction. The optimization problem (3) does not scale well for high dimensional decision space. For instance, when training a deep neural network, there are often more than millions of parameters to be optimized. According to (Fliege and Svaiter 2000), based on the KKT conditions, d^t=$-Q\lambda$, where λ is the optimal solution of the following quadratic problem, which is the dual form of Problem (3), and this d^t adheres to Lemma 1.

$$\begin{aligned} &\max_{\lambda} -\tfrac{1}{2}\lambda^T(Q^TQ)\lambda \\ &s.t.\ \ \textstyle\sum_{i=1}^{|\lambda|}\lambda_i = 1, \\ &\quad\ \lambda_i \geq 0, \forall i = 1, 2, \cdots, |\lambda|. \end{aligned} \tag{6}$$

In practice, when the model is tolerable-fair, we set $Q = \nabla L(\omega^t)$, and thus $\lambda \in \mathbb{R}^m$, leaving us with an m-dimensional problem to solve; When the model is unfair, we set Q=concat$(\nabla L(\omega^t), \nabla L(\omega^t)h^t) \in \mathbb{R}^{n\times(m+1)}$ to yield d^t by solving an $(m+1)$-dimensional problem. Ultimately, we scale the length of $\|d^t\|$. The reason is that if we directly employ a method similar to FedSGD (McMahan et al. 2017) to calculate a direction, i.e., d_r=$-\frac{1}{|S_t|}\sum_i^{|S_t|} g_i$, $\|d^t\|\leq\|d_r\|$ is always true when Q=$\nabla L(\omega^t)$. To prevent the convergence from being affected by the norm of d^t, we scale d^t by d^t=σd^t, where σ=$\|d_r\|/\|d^t\|, d^t \neq \vec{0}$.

Historical Fairness. Due to clients dropping out and partial selection of clients at each round, not all clients could participate in FL, and there would be a sample bias (Wang et al. 2021). In this case, at round t, the server can only acquire the local gradients uploaded by the online clients S_t. Thus, Problem (2) is decomposed into the following dynamic subproblem, which is $|S_t|$-dimensional, 0<$|S_t|$<m.

$$\min_{\omega}\ (L_1(\omega), L_2(\omega), \cdots, L_{|S_t|}(\omega)). \tag{7}$$

If we apply the above approach to generate a direction for minimizing Problem (7), i.e., by setting Q = concat$(\nabla L_1(\omega^t), \cdots, \nabla L_{|S_t|}(\omega^t))$, and solve Problem (6), it may not be a descent direction of those clients who were online at round $t-1$ but are absent now. Thus, the model will be unstable when clients drop frequently. To address this issue, we estimate their true gradients by their gradients at round $t-1$ (denote it as g_{old}), and calculate a common descent direction that can improve the model performance in both the existing clients and those newly absent clients. In this regard, we can concatenate g_{old} to the matrix Q and solve Problem (6) to gain λ, and finally calculate $d^t = -Q\lambda$ to obtain a historical fair descent direction.

Besides, even if the model may still lie in the tolerable-fair area, the performance for a newly returned client may be significantly poorer than previously. In this situation, we set a reference objective vector L_i^R for the present objective $L_i(\omega^t)$. Once $\exists i \in S_t, L_i(\omega^t) > L_i^R$, the fairness-enhanced strategy will be activated. At round 0, L_i^R is initialized to the loss L_i^1. At round t>0, we update L_i^R=$\frac{L_i^R\times t+L_i^t}{t+1}$ if L_i^t<L_i^R.

Enhance Robustness. Gradient-based Federated learning can be easily affected by clients' attacks. For example:

1. (Zero-update attack). When client i sends a zero-norm gradient to the server, some previous methods, e.g., FedMGDA+ will trap into the weak Pareto stationarity (i.e., $\nabla L_i(\omega)$=$\vec{0}$) and the obtained direction will always be $\vec{0}$.
2. (Extreme norm attack). Some clients may upload a gradient with a too-large or too-small norm to disturb FL.

Hence, we propose a strategy to lessen the blow of such attacks and strengthen resilience. Firstly, we drop those clients who upload a 0-norm gradient to the server. Although it may temporarily sacrifice the model's performance on those clients, it can prevent the model from being poor weak Pareto stationary, allowing FL to continue training to create a superior model. The second step is to reallocate the norm of all received gradients to their average. The implementation is described in detail in Algorithm 1, line 5 and 6.

3.3 Step Size Line Search

Even when the direction d^t is fair descent, if the step size is too large, the new model may become increasingly unfair or even fail to converge. As a result, we must look for a suitable step size. At each round t, after transmitting d^t to clients, FedMDFG follows the proposed Algirithm 2 to search η^t.

In Algorithm 2, we describe how to perform a step size η^t using a backtracking procedure. From the upper bound η_{ub} to the lower bound $(1/2)^s\eta/\sigma$, it stops updating $\eta^t = \eta^t/2$ if the stopping criterion, i.e., the Armijo condition (Nocedal and Wright 1999) (Algorithm 2, line 8) is satisfied. s is a given positive integer that controls the searching range of the step size and $\beta \in (0, 1)$ is a parameter of Armijo condition. In practice, we fix β to 10^{-4}. Note that when the model is unfair, it needs to consider an additional condition further: the angle between L and p is smaller than the previous. (See Algorithm 2, line 9). The lower bound is designed to prevent η^t from being too small, which can help jump out of the local optimum; the upper bound is designed to find a larger step size to increase the convergent rate. We set $\eta_{ub} = 2^s\eta$ for $S_{old} \subset S_t$ and $\eta_{ub} = \eta$ for $S_{old} \not\subset S_t$, i.e., some clients of the last round are offline at the current round. The reason is that if some clients drop, a step that is too large may impair the model's performance on those absent clients.

Every time η^t is updated, it will be sent to clients. Each client refreshes a temporary model by $\omega^t + \eta^t d^t$, computes its loss, and then delivers it to the server. The server will then determine whether η^t satisfies the stopping condition.

We denote the above process as the first stage. Furthermore, to increase global convergence in nonconvex cases, when the Armijo condition cannot be satisfied in the first stage, we start a second stage: choosing the biggest η^t from a history set H that the corresponding loss vector L satisfies the condition $\|L\|_1 < \|L(\omega^t)\|_1$. H is a set that stores all η^t searched in the first stage. In the end, if the above two stages cannot determine a step size, we select an η^t with the smallest ℓ^1-norm of the corresponding L from H.

In two ways, the line search procedure has low communication costs. Firstly, in this process, data transit between the server and clients only consists of float-type scalars, which need a small amount of bandwidth. Besides, it doesn't take too long to wait for clients to calculate losses since they

Algorithm 2: LineSearch

Input: $\sigma, S_t, S_{old}, s, \eta, p, L(\omega^t), g^t, d^t, f$
Output: η^t

$\eta_{ub} \leftarrow 2^s\eta$; ($\eta_{ub} \leftarrow \eta$ when $S_{old} \not\subset S_t$)
$\eta_{lb} \leftarrow (1/2)^s\eta/\sigma$.
Initialize $\eta^t \leftarrow \eta_{ub}$, $H = \{\}$, and a flag $stop \leftarrow False$.
while $\eta^t \geq \eta_{lb}$ **do**
  Send η^t to all clients S_t.
  Each client i builds a temporary model by $\omega^t + \eta^t d^t$, and then evaluates its loss L_i and uploads them.
  $L \leftarrow (L_1, \cdots, L_{|S_t|})$, and Store (η^t, L) in H.
  if $L_i(\omega^t) \geq L_i - \beta\eta^t g^t d^t, \forall i \in S_t$ **then**
    if $f = True$ or $\varphi(L, p) < \varphi(L(\omega^t), p)$ **then**
      $stop \leftarrow True$.
      break
    end if
  end if
  $\eta^t \leftarrow \eta^t/2$.
end while
if $stop = False$ **then**
  Select the biggest η^t in H that the corresponding L satisfies $\|L\|_1 < \|L(\omega^t)\|_1$.
end if
if $stop = False$ **then**
  Select the η^t from H that the corresponding L has the smallest value of ℓ^1-norm.
end if

don't need to compute the model's gradients and instead calculate the loss through the forward pass, which isn't time-consuming with parallel computation techniques (Li et al. 2020a). The matrix multiplication, which is the fundamental operation of the forward pass, has a time complexity of O(1) (Cai et al. 2016) in some of the most up-to-date devices, such as a fully memristor-implemented neural network (Yao et al. 2020), allowing it to swiftly acquire the model loss.

3.4 Analysis

Convergence. Assume that all clients are selected at each round, the convergence of FedMDFG is guaranteed by the fair descent direction d^t and a proper step size η^t. When η^t satisfies the Armijo condition, according to Theorem 2, which is inspired by (Fliege and Svaiter 2000), FedMDFG converges to a Pareto critical point of Problem (2). In another case, the learning rate η is set too large such that η^t satisfies the condition $\|L(\omega^t + \eta^t d^t)\|_1 < \|L(\omega^t)\|_1$. Since $(|L(\omega^t)||_1)_t$ is decreasing and bounded by 0, in accordance with Monotone Convergence Theorem (Rudin et al. 1976), FedMDFG can converge to a local optimum of Problem (1). The detail proof can be seen in Appendix A.2.

Theorem 2. Suppose that the Armijo condition in the step size line search process is satisfied, then every accumulation point of the sequence $(w^t)_t$ produced by FedMDFG is a Pareto critical point.

Fairness. In Section 3.2, we have discussed how the fair-driven objective can help get a fair descent direction to guide the model fairer. Note that we only activate this extra objective when the model is not tolerable-fair to prevent the local objectives from being the same. Or the model is easily trapped into a local optimum since there doesn't exist a direction or a proper step size that can drive the model fairer, and thus it will affect the model to be trained further. We also specified that the step size could not be too large; otherwise, the new model would become excessively unfair. Hence, we design the step size line search to find a suitable step size.

4 Experiments

4.1 Experimental Setup

All experiments are implemented on a server with Intel(R) Xeon(R) Silver 4216 CPUs and 2 NVidia(R) 3090 GPUs.

Datasets and Models. We evaluate the performance of FedMDFG on four public datasets: CIFAR-10, CIFAR-100 (Krizhevsky and Hinton 2009), MNIST(LeCun et al. 1998), and Fashion MNIST (FMNIST) (Xiao, Rasul, and Vollgraf 2017), where the training/testing data are already split. To simulate different local data distributions, we design three scenarios to assign data for clients. (1) Normal: we follow (McMahan et al. 2017) to sort all data records based on their classes, divide them into 200 shards, and assign each of 100 clients 2 shards randomly without replacement. (2) Mutex: Each client only has all the data of one class of the dataset, i.e., client i only has all the data of class j. (3) Unbalanced cases for CIFAR-10 and FMNIST. It is an extension of the mutex case, where 10 clients are randomly split into five groups: group 1 has one client; Each of group $2, 3, 4$ has two clients; The last group has 3 clients. Then, we aggregate all clients from each group into a new client. We utilize CNN for MNIST, CIFAR-10, and CIFAR-100, following the setting in (Wang et al. 2021). Besides, we adopt Multilayer perceptron (Popescu et al. 2009) for FMNIST.

Baselines and Hyper-parameters. We compare with FedAvg (McMahan et al. 2017), the traditional method, and FL algorithms that address fairness, including AFL (Mohri, Sivek, and Suresh 2019), q-FedAvg (Li et al. 2020c), TERM (Li et al. 2021a), FedFV (Wang et al. 2021), Ditto (Li et al. 2021b), and FedMGDA+ (Hu et al. 2022). Based on the code supplied by their authors, our platform directly rewrites the code of the fair FL techniques that are being compared. Table 1 list the hyper-parameters of various techniques we verify. The first one of each parameter set is regarded as the default for each algorithm. All clients utilize Stochastic Gradient Descent (SGD) on local datasets with the learning rate $\eta \in \{0.01, 0.05, 0.1\}$ and decay of 0.999 per round, and take the best performance of each method in comparison. We average the results in 5 runs with different random seeds.

4.2 Accuracy and Fairness

We first test the performance of FedMDFG in the normal, mutex, and unbalanced cases on CIFAR-10 with batch size 50. 10%, 50%, and 60% of clients are randomly sampled at each communication round, respectively. Table 2 lists the mean (and the fairness indicator mentioned in Section 3.1) of test accuracy on all clients over 2000 rounds. FedMDFG

Method	Hyper-parameters
AFL	$\gamma_\lambda \in \{0.1, 0.01\}$
qFedAvg	$q \in \{0.1, 1.0\}$
FedFV	$\alpha \in \{0.1, 0.2\}, \tau \in \{1, 2\}$
TERM	$t \in \{1, 5\}$
Ditto	$\lambda \in \{0.1, 1.0\}$
FedMGDA+	$\epsilon \in \{0.1, 1.0\}$
FedMDFG	$\theta \in \{\pi/16, \pi/32, \pi/180\}, s \in \{5, 3, 1\}$

Table 1: Hyper-parameters of different methods

	Normal	Mutex	Unbalanced
FedAvg	.683(.202)	.323(.728)	.547(.473)
AFL$\|_{\gamma_\lambda=0.01}$	.681(.141)	.328(.589)	.526(.538)
AFL$\|_{\gamma_\lambda=0.1}$	.689(.139)	.278(.637)	.525(.374)
qFedAvg$\|_{q=0.1}$	.667(.174)	.408(.339)	.558(.451)
qFedAvg$\|_{q=5.0}$	.446(.182)	.301(.291)	.420(.535)
FedFV$\|_{a=0.1,\tau=1}$	.684(.177)	.374(.582)	.561(.307)
FedFV$\|_{a=0.1,\tau=2}$	.689(.171)	.347(.600)	.554(.259)
FedFV$\|_{a=0.2,\tau=2}$	.685(.183)	.330(.606)	.532(.432)
TERM$\|_{t=1}$	.529(.176)	.338(.297)	.579(.263)
TERM$\|_{t=5}$	.520(.156)	.309(.389)	.557(.261)
Ditto$\|_{\lambda=0.1}$	.609(.201)	.310(.662)	.537(.470)
Ditto$\|_{\lambda=0.5}$	.614(.200)	.290(.716)	.534(.355)
FedMGDA+$\|_{\epsilon=0.1}$	.507(.201)	.324(.463)	.522(.286)
FedMGDA+$\|_{\epsilon=1.0}$	.505(.201)	.324(.462)	.524(.367)
FedMDFG$\|_{\theta=\frac{\pi}{180},s=5}$	.738(**.093**)	.657(**.157**)	.641(**.173**)
FedMDFG$\|_{\theta=\frac{\pi}{32},s=5}$	**.746**(.100)	.690(.165)	.659(.174)
FedMDFG$\|_{\theta=\frac{\pi}{16},s=5}$	.743(.103)	.685(.209)	**.661**(.181)
FedMDFG$\|_{\theta=\frac{\pi}{16},s=3}$	.743(.101)	**.691**(.203)	.659(.178)
FedMDFG$\|_{\theta=\frac{\pi}{16},s=1}$	.742(.101)	.689(.202)	.660(.195)

Table 2: The average test accuracy of all clients (and the fairness indicator) in normal, mutex, and unbalanced case on CIFAR-10 with batch size 50.

improves the average test accuracy to 74.6%, 69.1%, and 66.1% in the above cases, which is much better than other methods. Besides, compared to the second-best method, FedMDFG obtains much fairer models where the fairness indicator decreased by 33.1%, 46.0%, and 33.2% for three cases, respectively. According to the findings of various hyper-parameters, FedMDFG can obtain a fairer model by setting a smaller tolerable-fair angle θ.

We then compare the best 10%, the average, and the worst 10% test accuracy across clients of different methods in the mutex case on CIFAR-100 with batch size 200 over 5000 rounds. Since each client only has access to one class of the CIFAR-100, it's challenging to treat each client fairly. FedMDFG not only has the highest average model performance on clients but also shields the worst performance as per Fig. 3. The lowest 10% test accuracy of other algorithms, in contrast, are close to 0.

Additionally, we model a situation where there is only one client online in FL for a long time before other clients come in. Firstly, a client from the mutex case on CIFAR-10 is randomly selected to train the model over 50 epochs, resulting

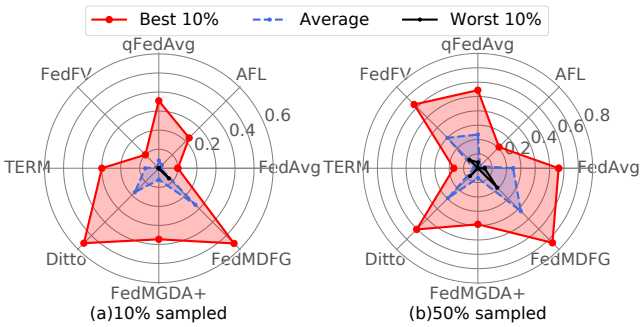

Figure 3: The best 10%, the average, and the worst 10% test accuracy of 100 clients in the mutex case on CIFAR-100. (a) 10% clients randomly sampled per round. (b) 50% sampled.

	Ave	Fairness	Worst	Best
Overfitting	.100	1.25	.000	1.00
FedAvg	.325	.467	.055	.580
AFL	.100	1.25	.000	1.00
qFedAvg	.472	.357	.178	.742
FedFV	.309	.235	.173	.432
TERM	.396	.233	.209	.504
Ditto	.369	304	.203	.542
FedMGDA+	.100	1.25	.000	1.00
FedMDFG	**.660**	**.075**	**.565**	.714

Table 3: The average, the fairness indicator, the worst, and the best test accuracy over 3000 rounds in the mutex case on CIFAR-10 with batch size 200 and local epochs $E = 1$.

in a training loss that is virtually zero and a test accuracy of 100% (overfitting). After that, the rest night clients join FL. We compare the final performance of algorithms over 3000 communication rounds. From Table. 3, we observe that AFL and FedMGDA+ fail to train the FL model since their final performances are the same as those of the overfitting model. FedMDFG can efficiently address the local-overfitting issue and obtain a fairer model with higher average test accuracy while protecting the worst local performance.

4.3 Efficiency and Convergence

We compare the convergent efficiency of different algorithms in the mutex case on CIFAR-100 and MNIST with local epochs $E = 1$. All methods are tuned to their best performance. 100% of clients are selected at each round. Fig. 4 shows part of the results, while the methods with poor performance are hidden. Full results are available in Appendix B.2. We can observe that FedMDFG converges much faster than others and keeps the model fairer as well. Some previous methods, e.g., FedAvg (E=10) and FedFV, can easily suffer significant performance reductions on clients as they cannot ensure a common descent direction for model updating. If we tune a smaller η, its convergence would be much slower. For FedAvg E=10, it takes much more local computation resources and time since each client needs to compute gradients repeatedly, which lengthens the communication lag. Instead of the local training, FedMDFG performs the

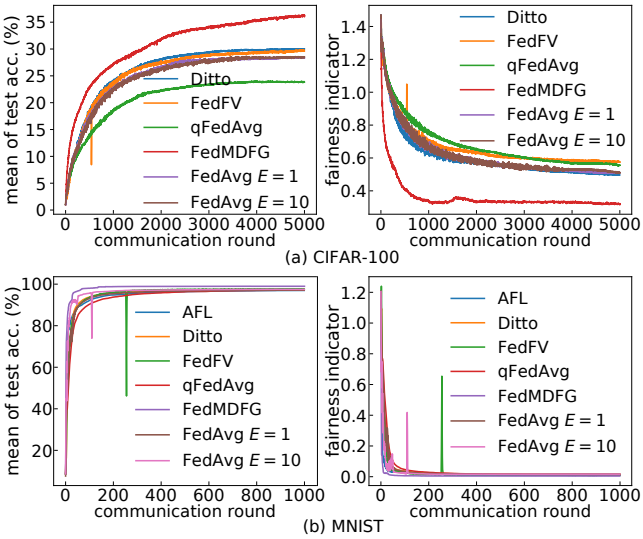

Figure 4: The mean (left) and the fairness indicator (right) of the test accuracy across all clients in the mutex case on (a) CIFAR-100 and (b) MNIST with batch size 200 and $E = 1$.

	clean	A1	A2	A3
FedAvg	.859(.100)	.354(.665)	.103(1.10)	.546(.330)
AFL	.834(.087)	.103(1.09)	.103(1.10)	.380(.438)
qFedAvg	.855(.106)	.099(1.11)	.486(.563)	.549(.323)
FedFV	.862(.091)	.454(.487)	.103(1.10)	.563(.385)
TERM	.672(.201)	.278(.734)	.637(.090)	.601(.267)
Ditto	.860(.095)	.371(.634)	.089(1.14)	.584(.379)
FedMGDA+	.714(.216)	.674(.272)	.714(.216)	.103(.109)
FedMDFG	**.875(.065)**	**.875(.080)**	**.876(.066)**	**.866(.064)**

Table 4: The average test accuracy (and the fairness indicator) under clients' attacks in the normal case on FMNIST.

low-communication-cost step size searching to find a proper step size at each round to train the model fairly and quickly, since the direction is fair descent.

4.4 Robustness

Three kinds of attacks are built for testing the robustness of algorithms on FMNIST: (A1) Random updates: Malicious clients transmit random-zero-mean Gaussian parameters (or gradients, for gradient-based FL) (Li et al. 2021b); (A2) Enlarged update: When client i tries to dominate or disturb the FL, it sends a fake gradient $100g_i$ (or $100(\omega_i^{t+1}-\omega^t)+\omega^t$ if the method is non-gradient-based) to the server. (A3) Zero-update: client i uploads $g_i=\vec{0}$ (or $\omega^{t+1}=\omega^t$ for non-gradient-based methods) to disturb FL. All 100 clients participate in FL at each round, 10% of which are dishonest. Table. 4 shows that FedMDFG is resilient under these attacks, while other algorithms suffer from variant reductions on the average test accuracy and fairness. By determining a fair descent direction and ensuring a suitable step size, FedMDFG can protect the performance of honest clients well.

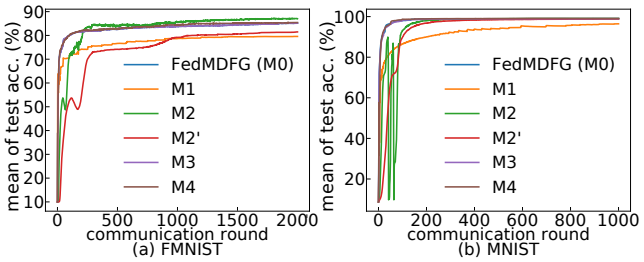

Figure 5: Average test accuracy of clients in the mutex case on (a) FMNIST and (b) MNIST in the ablation experiment. 100% of clients are online at each round.

4.5 Ablation Experiments

To observe the effect of each part of FedMDFG (M0), we compare it with four modified algorithms: (M1) Replace the fair descent direction with d_r gotten by FedSGD (McMahan et al. 2017); (M2) Skip the step size line search and adopt η as the step size; (M3) Setting β (a parameter of Armijo condition) to 0. In Section 3.3 we fix it to 10^{-4}; (M4) Setting $\beta = 0.5$. The comparison findings are depicted in Fig. 5, which demonstrates that without ensuring the direction is fair descent at each round, M1 performs much worse than FedMDFG. Besides, although M2 can finally get a model with slightly higher average test accuracy, it's highly unstable because of the improper step size it uses at each round. If we tune η to a smaller one (denoted as M2′) to stabilize the performance, the convergence would be considerably slower. The performances of M0, M3, and M4 are similar (The curve of M0 and M3 in Fig. 5 (a) are overlapped), indicating that β can be fixed.

5 Conclusion and Future Work

We propose FedMDFG, a federated learning algorithm with multi-gradient descent and fair guidance to enhance fairness by addressing two direct causes of unfairness: using an unfair direction and an improper step size to update the model. We show how well FedMDFG can obtain a fair descent direction and how to confirm a suitable step size through the low-communication-cost step size line search. Empirical results demonstrate that FedMDFG is robust under a diverse set of attacks and outperforms the SOTA algorithms in terms of fairness and convergence. There are a number of intriguing future research topics, such as building an advanced step size searching strategy to accelerate the convergence and taking into account popular concerns such as privacy.

Acknowledgments

This paper is supported in part by the National Natural Science Foundation of China (Grant No. 72171206, No. 71931003, No. 72061147004, No. 72192805 and No. 62001412), in part by the National Key R&D Program of China (Grant No. 2018YFB1800800), and in part by the Guangdong Provincial Key Laboratory of Future Networks of Intelligence (Grant No. 2022B1212010001). This paper is also supported in part by the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS).

References

Abay, A.; Zhou, Y.; Baracaldo, N.; Rajamoni, S.; Chuba, E.; and Ludwig, H. 2020. Mitigating bias in federated learning. *arXiv preprint arXiv:2012.02447*.

Cai, R.; Ren, A.; Wang, Y.; Soundarajan, S.; Qiu, Q.; Yuan, B.; and Bogdan, P. 2016. A low-computation-complexity, energy-efficient, and high-performance linear program solver using memristor crossbars. In *2016 29th IEEE International System-on-Chip Conference*, 317–322.

Cho, Y. J.; Gupta, S.; Joshi, G.; and Yağan, O. 2020. Bandit-based communication-efficient client selection strategies for federated learning. In *2020 54th Asilomar Conference on Signals, Systems, and Computers*, 1066–1069. IEEE.

Deng, Y.; Kamani, M. M.; and Mahdavi, M. 2020. Distributionally robust federated averaging. *Advances in Neural Information Processing Systems*, 33: 15111–15122.

Désidéri, J.-A. 2012. Multiple-gradient descent algorithm (MGDA) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5-6): 313–318.

Fliege, J.; and Svaiter, B. F. 2000. Steepest descent methods for multicriteria optimization. *Mathematical methods of operations research*, 51(3): 479–494.

Fliege, J.; and Vaz, A. I. F. 2016. A method for constrained multiobjective optimization based on SQP techniques. *SIAM Journal on Optimization*, 26(4): 2091–2119.

Gebken, B.; Peitz, S.; and Dellnitz, M. 2019. On the hierarchical structure of Pareto critical sets. *Journal of Global Optimization*, 73(4): 891–913.

Hu, Z.; Shaloudegi, K.; Zhang, G.; and Yu, Y. 2022. Federated Learning Meets Multi-Objective Optimization. *IEEE Transactions on Network Science and Engineering*, 9(4): 2039–2051.

Huang, W.; Li, T.; Wang, D.; Du, S.; Zhang, J.; and Huang, T. 2022. Fairness and accuracy in horizontal federated learning. *Information Sciences*, 589: 170–185.

Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2): 1–210.

Kanaparthy, S.; Padala, M.; Damle, S.; and Gujar, S. 2022. Fair Federated Learning for Heterogeneous Data. In *5th Joint International Conference on Data Science & Management of Data*, 298–299.

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Li, S.; Zhao, Y.; Varma, R.; Salpekar, O.; Noordhuis, P.; Li, T.; Paszke, A.; Smith, J.; Vaughan, B.; Damania, P.; and Chintala, S. 2020a. PyTorch Distributed: Experiences on Accelerating Data Parallel Training. *Proceedings of the VLDB Endowment*, 13(12): 3005–3018.

Li, T.; Beirami, A.; Sanjabi, M.; and Smith, V. 2021a. Tilted Empirical Risk Minimization. In *International Conference on Learning Representations*.

Li, T.; Hu, S.; Beirami, A.; and Smith, V. 2021b. Ditto: Fair and Robust Federated Learning Through Personalization. In *ICML*, 6357–6368.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2020b. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2: 429–450.

Li, T.; Sanjabi, M.; Beirami, A.; and Smith, V. 2020c. Fair Resource Allocation in Federated Learning. In *8th International Conference on Learning Representations, ICLR-2020*. OpenReview.net.

Lyu, L.; Xu, X.; Wang, Q.; and Yu, H. 2020. Collaborative fairness in federated learning. In *Federated Learning*, 189–204. Springer.

Lyu, L.; Yu, J.; Nandakumar, K.; Li, Y.; Ma, X.; and Jin, J. 2019. Towards fair and decentralized privacy-preserving deep learning with blockchain. *arXiv preprint arXiv:1906.01167*, 1–13.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 1273–1282. PMLR.

Mohri, M.; Sivek, G.; and Suresh, A. T. 2019. Agnostic federated learning. In *International Conference on Machine Learning*, 4615–4625. PMLR.

Nocedal, J.; and Wright, S. J. 1999. *Numerical optimization*. Springer.

Popescu, M.-C.; Balas, V. E.; Perescu-Popescu, L.; and Mastorakis, N. 2009. Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8(7): 579–588.

Rudin, W.; et al. 1976. *Principles of mathematical analysis*, volume 3. McGraw-hill New York.

Wang, Z.; Fan, X.; Qi, J.; Wen, C.; Wang, C.; and Yu, R. 2021. Federated Learning with Fair Averaging. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 1615–1623. IJCAI Organization.

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

Yao, P.; Wu, H.; Gao, B.; Tang, J.; Zhang, Q.; Zhang, W.; Yang, J. J.; and Qian, H. 2020. Fully Hardware-Implemented Memristor Convolutional Neural Network. *Nature*, 577(7792): 641–646.

Zhao, Z.; and Joshi, G. 2022. A Dynamic Reweighting Strategy For Fair Federated Learning. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 8772–8776. IEEE.