HISTORICAL SELF-REPAIR FOR FEDERATED LEARNING WITH DELAYED DETECTION OF NOISY LABELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated learning with noisy labels aims to train robust global models in federated settings where clients may hold noisy labels. Most existing methods follow a detection-and-correction paradigm: they identify noisy clients or samples during training and correct the detected noise through label correction, filtering, or robust optimization. However, this paradigm faces the problem of delayed detection, where some noisy samples are detected after multiple communication rounds. Although these samples can be corrected after being detected, their updates have already affected the global model during the earlier training step, leaving harmful historical influence on subsequent optimization. To address this problem, we propose FedHSR, a historical self-repair framework for robust model training. FedHSR maintains the global training trajectory and repairs the affected historical trajectory through two modules. Historical Replay reconstructs the historical updates of noisy clients based on the detected data, while Historical Response Calibration estimates the responses of clean clients to the model shift induced by historical repair without requiring them to repeat their historical training. We further establish theoretical bounds for both response estimation and trajectory repair. Experiments show that FedHSR effectively repairs the affected trajectory and consistently improves the performance of existing methods. Code is available at https://anonymous.4open.science/r/ICLR-59567.

1 INTRODUCTION

Federated learning (FL) is a privacy-preserving distributed learning framework that has been widely used in applications such as healthcare, mobile intelligence, and financial services McMahan et al. (2017); Kairouz & McMahan (2021); Rieke et al. (2020); Yang et al. (2019). In federated learning, each client trains on its local data and uploads model updates for global aggregation McMahan et al. (2017); Li et al. (2020b). Therefore, federated optimization is highly dependent on the quality of local labels Xu et al. (2022); Ji et al. (2024). However, local labels may be corrupted by annotation errors or ambiguous samples Wei et al. (2021), resulting in label noise that affects local training and further degrades the global model through federated aggregation Ke et al. (2022). Consequently, **federated learning with noisy labels** Xu et al. (2022); Wu et al. (2023) has become a fundamental problem in robust federated learning.

Existing methods Xu et al. (2022); Wu et al. (2023); Ji et al. (2024); Wen et al. (2026) for federated learning with noisy labels mainly follow two paradigms. The first is **detect-then-correct**. It separates training into a detection stage and a correction stage Xu et al. (2022); Wu et al. (2023). In the detection stage, the model identifies noisy clients or samples over several communication rounds. In the correction stage, the detected noise is handled through label correction, filtering, or robust training. This paradigm has achieved strong performance because the detection stage can provide relatively stable noise estimation. However, federated learning is dynamic Ruan et al. (2021); Bonawitz et al. (2019); Yan et al. (2024); Ribero et al. (2022), and clients unavailable during one detection stage may only be inspected in a later stage. To address this issue, some recent methods follow a **detect-and-correct** paradigm. They perform detection and correction during training, making them suitable for settings in which noisy samples are identified progressively Ji et al. (2024); Wen et al. (2026).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

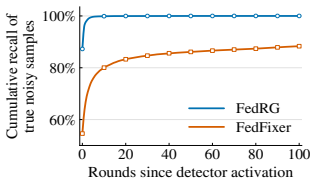


Figure 1: Cumulative noisy samples detected over training rounds.

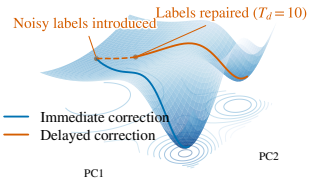


Figure 2: Training trajectories with immediate and delayed correction.

However, existing detection-and-correct methods face a **delayed detection** problem. Specifically, noisy samples are usually not identified immediately during the training steps. Figure 1 reports the number of detected noisy samples over communication rounds based on several state-of-the-art methods Ji et al. (2024); Wen et al. (2026). Noisy samples can remain undetected for dozens of communication rounds. We refer to this phenomenon as **delayed detection**. Such delayed detection allows undetected noisy samples to participate in local training for multiple rounds and affect the global model before they are corrected. Figure 2 compares the training trajectories of the delayed correction samples and immediate correction samples. Although correcting the samples after detection prevents further corrupted updates, the training trajectory still differs from the one produced by immediate correction. Therefore, delayed noise detection introduces historical influence that remains after sample correction and continues to affect subsequent federated optimization. This motivates us to repair the historical influence accumulated before noise detection.

To address this problem, we propose FedHSR, a **historical self-repair** framework for federated learning with delayed noise detection. Unlike existing methods that mainly correct detected noisy labels for subsequent training, FedHSR focuses on repairing the historical influence of noisy labels on the training trajectory. Specifically, FedHSR maintains the global model of each communication round at the server and reconstructs the trajectory from the first round in which a noisy sample participates in training. For the clients with noisy samples, we introduce the **Historical Replay** module to replay their historical training with the correction rule available after detection. For the remaining clean clients, we introduce the **Historical Response Calibration** module, which uses the observed historical trajectory to estimate their aggregate update response to the model shift induced by historical repair. Therefore, FedHSR reconstructs the repaired trajectory without retraining all historical clients. We further derive theoretical bounds for estimating the aggregate responses of clean clients and analyze how the estimation error affects the reconstructed trajectory.

Our main contributions are summarized as follows:

- We identify historical influence under delayed noise detection as an overlooked problem in federated learning, where noisy samples may affect the training trajectory for several rounds before being detected.
- We propose FedHSR, a historical self-repair framework for delayed noise detection. Historical Replay reconstructs the historical updates of noisy clients, while Historical Response Calibration avoids replaying all historical clean clients.
- We provide theoretical guarantees for estimating the aggregate responses of clean clients and the resulting trajectory repair. Extensive experiments further validate the existence of delayed detection and historical influence, and demonstrate the effectiveness of FedHSR under federated learning with noisy labels.

2 Related Work

2.1 Federated Learning

Federated learning enables distributed clients to collaboratively optimize a global model without sharing their raw data McMahan et al. (2017); Kairouz & McMahan (2021). Prior work improves its local optimization, aggregation, client selection, and communication efficiency Li et al. (2020b); Karimireddy et al. (2020); Wang et al. (2020); Lai et al. (2021). In practice, however, incorrect local annotations can distort both client optimization and the aggregated global model Wei et al. (2021); Ke et al. (2022), motivating federated learning robust to noisy labels.

2.2 Federated Learning with Noisy Labels

Existing methods identify noisy clients or samples and mitigate their effect through robust objectives, filtering, aggregation, or label correction Tsouvalas et al. (2024); Fang & Ye (2022); Jiang et al. (2022); Kim et al. (2022); Zhou & Wang (2024); Yang et al. (2022); Patrini et al. (2017). FedCorr, FedNoRo, and FedELC perform client-level detection followed by noise mitigation or correction Xu et al. (2022); Wu et al. (2023); Jiang et al. (2024), whereas FedFixer, FedDiv, FedClean, and FedRG more closely integrate sample identification with federated optimization Ji et al. (2024); Li et al. (2024); Jiang & Zhang (2025); Wen et al. (2026). However, these methods do not account for delayed detection, under which noisy samples may affect the global training trajectory for multiple rounds before being identified and corrected. Our work targets this historical influence of delayed detection and is complementary to their detectors and correction rules.

2.3 Relation to Federated Unlearning and Model Recovery

Federated unlearning and model recovery also reconstruct counterfactual models after undesirable influence has occurred Bourtoule et al. (2021). FedEraser recalibrates stored updates to remove a client, FedRecover estimates updates to recover from malicious clients Liu et al. (2021); Cao et al. (2023), and FedNLU unlearns noisy-label influence from the current model before resuming training Jin et al. (2026). Their target differs from ours: unlearning approximates training without the target data, while FedHSR reconstructs the historical trajectory under a corrected training rule, retaining samples that remain useful after relabeling or reweighting.

3 Method

3.1 Problem Formulation

We consider a standard federated learning system with a central server and K clients under full participation McMahan et al. (2017). At communication round t, each client k trains from the current global model $\mathbf{w}_g^{t-1}$ and produces a local update Δ_k^t. The server aggregates these updates as $\mathbf{w}_g^t = \mathbf{w}_g^{t-1} + \sum_{k=1}^K p_k^t \Delta_k^t$, where p_k^t denotes the aggregation weight of client k. Federated learning repeats this process over T communication rounds and produces a sequence of global models $\mathbf{w}_g^0, \mathbf{w}_g^1, \ldots, \mathbf{w}_g^T$.

Let $\mathcal{A}$ denote the set of clients containing the noisy samples, and let t_0 be the earliest round in which these samples affect training. Their noisy labels change the local updates of clients $a \in \mathcal{A}$ and therefore change the aggregated global model $\mathbf{w}_g^{t_0}$. The following rounds then continue training from this affected model and produce the trajectory $\mathbf{w}_g^{t_0}, \mathbf{w}_g^{t_0+1}, \ldots$. In contrast, if these samples had carried their correct labels from their first participation, the same federated training would produce the clean trajectory $\bar{\mathbf{w}}_g^{t_0}, \bar{\mathbf{w}}_g^{t_0+1}, \ldots$. Thus, the noisy samples cause the factual trajectory to gradually deviate from the clean trajectory.

Due to delayed noise detection, these samples may not be identified until round T_d. Therefore, the historical trajectory $\mathbf{w}_g^{t_0}, \ldots, \mathbf{w}_g^{T_d}$ has already been affected by the noisy labels. Correcting the samples at T_d prevents further corrupted updates, but continuing training from the current model does not repair the affected historical trajectory. Our goal is therefore to reconstruct a repaired trajectory $\hat{\mathbf{w}}_g^{t_0}, \ldots, \hat{\mathbf{w}}_g^{T_d}$ such that $\hat{\mathbf{w}}_g^{T_d}$ is as close as possible to the clean model $\bar{\mathbf{w}}_g^{T_d}$. The most direct solution is to return to round t_0 and retrain the complete federated process with the corrected labels. However, such historical retraining requires retaining or recovering the training states and rerunning all historical client training, incurring substantial storage and computational costs.

3.2 Historical Self-Repair

The deviation between the affected trajectory and the clean trajectory comes from two sources. First, for the noisy clients, the mislabeled samples change their local update in each affected round, causing the aggregated global model to move away from the clean trajectory. Second, once the global model has been affected by the noisy update, the clean clients in subsequent rounds also start their local training from an incorrect global model. As a result, the historical deviation is jointly caused by the direct error from the noisy client and the induced update error from the clean clients.

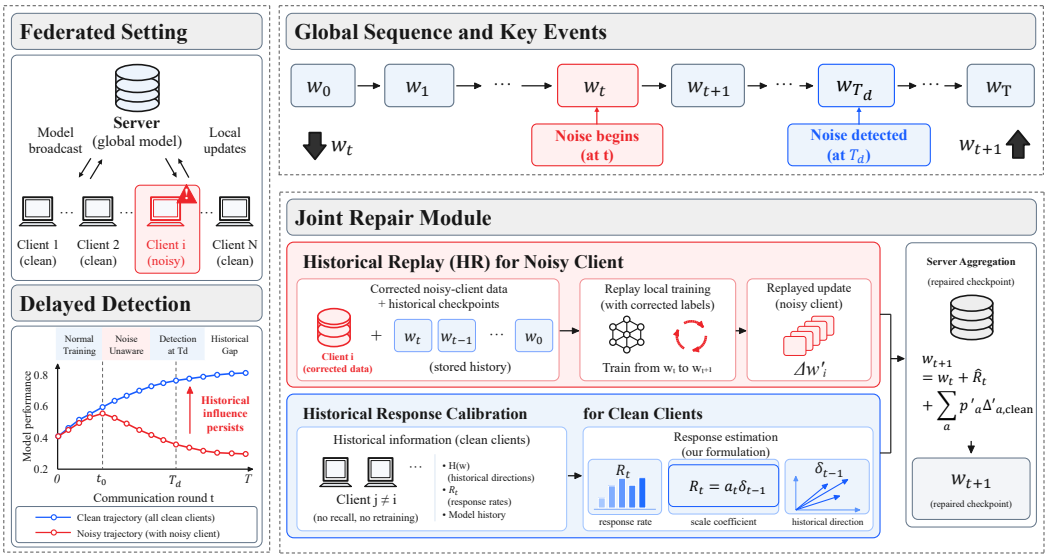


Figure 3: Overview of FedHSR. Historical Replay reconstructs corrected noisy client updates, while Historical Response Calibration estimates clean client responses to the repaired model shift. Their aggregation repairs the affected historical trajectory.

To repair these two sources of deviation, FedHSR maintains the historical global model trajectory $\mathcal{B}^T = \{\mathbf{w}_g^0, \mathbf{w}_g^1, \ldots, \mathbf{w}_g^T\}$ at the server. When a noisy sample is detected, FedHSR traces back to the round when it first affected training and reconstructs the historical trajectory round by round. For the noisy client, we introduce **Historical Replay** to reproduce its clean local updates using the corrected labels. For the clean clients, we introduce **Historical Response Calibration** to estimate how their updates should change under the shift of the repaired global model. These two components jointly reconstruct the repaired trajectory toward the clean trajectory.

3.2.1 Historical Replay

We first repair the direct deviation introduced by the noisy clients. Since the server stores the historical global model $\mathbf{w}_g^{t-1}$, each client $a \in \mathcal{A}$ can reconstruct its original noisy update, denoted as $\Delta_{a,\text{noisy}}^t(\mathbf{w}_g^{t-1})$. Once the correct supervision is available, each client $a \in \mathcal{A}$ can reconstruct the clean update $\Delta_{a,\text{clean}}^t(\hat{\mathbf{w}}_g^{t-1})$ that should have been produced in the clean trajectory. In realistic settings, perfectly clean supervision may be unavailable. FedHSR can instead use an existing noise detection and correction method, such as FedNoRo or FedRG Wu et al. (2023); Wen et al. (2026), to generate the repaired local updates for Historical Replay.

We then directly replace the historical contributions of the noisy clients in the global model:

$$\widetilde{\mathbf{w}}_g^t = \hat{\mathbf{w}}_g^{t-1} + \left(\mathbf{w}_g^t - \mathbf{w}_g^{t-1}\right) - \sum_{a\in\mathcal{A}} p_a^t \Delta_{a,\text{noisy}}^t(\mathbf{w}_g^{t-1}) + \sum_{a\in\mathcal{A}} p_a^t \Delta_{a,\text{clean}}^t(\hat{\mathbf{w}}_g^{t-1}). \tag{1}$$

Here, $\widetilde{\mathbf{w}}_g^t$ denotes the model obtained by starting from the repaired model $\hat{\mathbf{w}}_g^{t-1}$ and replacing the original historical contributions of the noisy clients with their corrected updates.

3.2.2 Historical Response Calibration

Through the Historical Replay, FedHSR repairs the global model in round $t-1$ from $\mathbf{w}_g^{t-1}$ to $\hat{\mathbf{w}}_g^{t-1}$. However, the updates of the clean clients in round t also change when they start from the repaired global model. In the original historical training, a clean client k produces the update $\Delta_k^t(\mathbf{w}_g^{t-1})$, while in the repaired trajectory, it should produce $\Delta_k^t(\hat{\mathbf{w}}_g^{t-1})$. Although the local data remain unchanged, these two updates are generally different because their starting global models are different.

For clarity, we denote the aggregate update of the clean clients in one round by $H(\mathbf{w}) = \sum_{k\notin\mathcal{A}} p_k \Delta_k(\mathbf{w})$. The affected trajectory contains the update $H(\mathbf{w}_g^{t-1})$. The repaired trajectory requires $H(\hat{\mathbf{w}}_g^{t-1})$. We define the model shift as $\boldsymbol{\delta}_{t-1} = \hat{\mathbf{w}}_g^{t-1} - \mathbf{w}_g^{t-1}$, equivalently, $\hat{\mathbf{w}}_g^{t-1} = \mathbf{w}_g^{t-1} + \boldsymbol{\delta}_{t-1}$. The corresponding update shift is

$$R_t = H(\hat{\mathbf{w}}_g^{t-1}) - H(\mathbf{w}_g^{t-1}) = H(\mathbf{w}_g^{t-1} + \boldsymbol{\delta}_{t-1}) - H(\mathbf{w}_g^{t-1}). \tag{2}$$

Here, the model shift $\boldsymbol{\delta}_{t-1}$ follows from the repaired result of the previous round. The original aggregate update of the clean clients is recovered by Historical Replay as $H(\mathbf{w}_g^{t-1}) = (\mathbf{w}_g^t - \mathbf{w}_g^{t-1}) - \sum_{a\in\mathcal{A}} p_a^t \Delta_{a,\text{noisy}}^t(\mathbf{w}_g^{t-1})$. Therefore, the remaining task is to estimate the change in the known update $H(\mathbf{w}_g^{t-1})$ induced by the known model shift $\boldsymbol{\delta}_{t-1}$.

Directly computing this change requires retraining and aggregating all historical clean clients from the repaired model. Therefore, we approximate it from the observed historical trajectory. For a sufficiently differentiable $H(\cdot)$, its Taylor expansion around the original historical model is

$$\begin{aligned} R_t &= H(\mathbf{w}_g^{t-1} + \boldsymbol{\delta}_{t-1}) - H(\mathbf{w}_g^{t-1}) \\ &= J_H(\mathbf{w}_g^{t-1})\boldsymbol{\delta}_{t-1} + \frac{1}{2!} D^2 H(\mathbf{w}_g^{t-1})[\boldsymbol{\delta}_{t-1}, \boldsymbol{\delta}_{t-1}] \\ &\quad + \frac{1}{3!} D^3 H(\mathbf{w}_g^{t-1})[\boldsymbol{\delta}_{t-1}, \boldsymbol{\delta}_{t-1}, \boldsymbol{\delta}_{t-1}] + \cdots . \end{aligned} \tag{3}$$

Here, J_H is the Jacobian of the aggregate operator of the clean clients and $D^q H[\cdot]$ denotes its qth directional derivative. The first term is linear in $\boldsymbol{\delta}_{t-1}$, whereas every subsequent term contains at least two factors of the repair displacement. Consequently, within a sufficiently local repair region, these higher order terms decay faster than the first order term and can be neglected, yielding $R_t \approx J_H(\mathbf{w}_g^{t-1})\boldsymbol{\delta}_{t-1}$. The remaining Jacobian is a high dimensional matrix that cannot be constructed without differentiating or rerunning the clean clients. Motivated by scalar secant approximations in spectral gradient methods Barzilai & Borwein (1988); Dai et al. (2019), we approximate it by $a_t\mathbf{I}$ and estimate a_t from two consecutive historical states:

$$a_t = \frac{\left\langle H(\mathbf{w}_g^{t-1}) - H(\mathbf{w}_g^{t-2}), \mathbf{w}_g^{t-1} - \mathbf{w}_g^{t-2} \right\rangle}{\left\| \mathbf{w}_g^{t-1} - \mathbf{w}_g^{t-2} \right\|_2^2}, \qquad \widehat{R}_t = a_t \boldsymbol{\delta}_{t-1}, \qquad R_t = \widehat{R}_t + \boldsymbol{\varepsilon}_t. \tag{4}$$

Here, a_t summarizes the locally observed response rate, while $\boldsymbol{\varepsilon}_t$ collects the higher order Taylor terms and the error caused by replacing the local Jacobian with a scalar. It is an analysis residual rather than an additional constant estimated by FedHSR.

We then bound this residual. Before presenting the analysis, we introduce a standard assumption on the objective of the clean clients underlying $H(\cdot)$. This assumption is widely adopted in the theoretical analysis of federated optimization Li et al. (2020c); Karimireddy et al. (2020); Yuan & Ma (2020).

Assumption 1 (Local Strong Convexity and Smoothness). Within the local repair region of round t, the aggregate objective $F(\cdot)$ of the clean clients is twice differentiable and satisfies $\mu_t\mathbf{I} \preceq \nabla^2 F(\mathbf{w}) \preceq L_t\mathbf{I}$, where $0 < \mu_t \leq L_t$ denote the lower and upper bounds of the local curvature, respectively.

To derive the response estimation bound, we further write the local update as $\Delta_k(\mathbf{w}) = \eta\Delta'_k(\mathbf{w})$, where η is the local learning rate and $\Delta'_k(\mathbf{w})$ denotes the corresponding update direction without the learning rate scaling. Under standard gradient based local optimization, $\Delta'_k(\mathbf{w}) = -\nabla F_k(\mathbf{w})$. Therefore, the aggregate update of clean clients can be written as $H(\mathbf{w}) = -\eta\nabla F(\mathbf{w})$, where $F(\mathbf{w}) = \sum_{k\notin\mathcal{A}} p_k F_k(\mathbf{w})$.

Proposition 1 (Stability of Scalar Response Approximation). Under Assumption 1 and $\|\mathbf{w}_g^{t-1} - \mathbf{w}_g^{t-2}\|_2 > 0$, the residual in Eq. equation 4 satisfies

$$\left\| R_t - \widehat{R}_t \right\| \leq \eta(L_t - \mu_t) \left\| \boldsymbol{\delta}_{t-1} \right\|. \tag{5}$$

The proof is provided in Appendix B.1.

Proposition 1 provides a stability bound for the scalar response approximation: its error is controlled by the local curvature range $L_t - \mu_t$ and the magnitude of the repair displacement. It does not

establish the optimality of the historical secant among all scalar coefficients. We use the historical secant because it provides a round specific coefficient from the observed trajectory without requiring the curvature bounds or retraining the clean clients. Its practical benefit over a fixed coefficient is examined in Table 3. We finally update the repaired global model as $\hat{\mathbf{w}}_g^t = \widetilde{\mathbf{w}}_g^t + \widehat{R}_t$.

Some federated settings select only a subset of clients in each communication round. For FedHSR, the resulting aggregate update of the clean clients may vary across rounds. Historical Response Calibration can estimate the response from clients shared by adjacent rounds. The remaining clean clients can be retrained from the repaired model. The same response estimation analysis applies to the shared clients. We also verify the feasibility of this strategy under partial participation.

The complete procedure is given in Algorithm 1 in Appendix A.

3.3 Trajectory Repair Analysis

In this section, we analyze why the proposed sequential repair can reconstruct a trajectory close to the clean trajectory. Proposition 1 bounds the response estimation error introduced at each individual round. As the historical trajectory is reconstructed sequentially, the error from each repaired round may further propagate through the subsequent federated updates. We therefore analyze the accumulated error over the complete repair process.

We define the clean federated transition at round t as $\Phi_t(\mathbf{w}) = \mathbf{w} + \sum_{a\in\mathcal{A}} p_a^t \Delta_{a,\text{clean}}^t(\mathbf{w}) + H(\mathbf{w})$, such that the clean trajectory satisfies $\bar{\mathbf{w}}_g^t = \Phi_t(\bar{\mathbf{w}}_g^{t-1})$. To characterize how an existing model deviation propagates across rounds, we assume that $\Phi_t(\cdot)$ is locally Lipschitz within the historical repair region: $\|\Phi_t(\mathbf{w}) - \Phi_t(\mathbf{w}')\| \le \kappa_t \|\mathbf{w} - \mathbf{w}'\|$. Such local smoothness and Lipschitz stability conditions are commonly adopted in the theoretical analysis of federated optimization Li et al. (2020b); Gorbunov et al. (2021). Intuitively, κ_t characterizes how much an existing trajectory deviation can be amplified after one round of clean federated training.

In practice, noise detection and correction cannot be guaranteed to be perfectly accurate. To account for this uncertainty, let $\boldsymbol{\xi}_t$ denote the difference between the resulting transition at round t and the ideal transition with exact detection and clean replay. Suppose each client level detection and repair decision succeeds with probability at least q_t, where a noisy client is detected and repaired and a clean client is left unchanged. Assume that a failed decision on client k changes the aggregate transition by at most $p_k^t B_{k,t}$. The sample level form follows by indexing sample contributions instead of clients.

Theorem 1 (Historical Trajectory Repair). Starting from the same global model before the detected noise first affects training, $\hat{\mathbf{w}}_g^{t_0-1} = \bar{\mathbf{w}}_g^{t_0-1}$, under the assumptions above, the reconstructed model at the detection round T_d satisfies

$$\mathbb{E}\left\|\hat{\mathbf{w}}_g^{T_d} - \bar{\mathbf{w}}_g^{T_d}\right\| \le \sum_{t=t_0}^{T_d} \left(\prod_{j=t+1}^{T_d} \kappa_j\right) \left[\eta(L_t - \mu_t)\mathbb{E}\left\|\boldsymbol{\delta}_{t-1}\right\| + (1-q_t)\sum_{k=1}^{K} p_k^t B_{k,t}\right]. \tag{6}$$

The proof is provided in Appendix B.2.

Theorem 1 separates the final trajectory deviation into the response estimation error and the error introduced by unsuccessful detection or correction. At round t, Historical Response Calibration contributes at most $\eta(L_t - \mu_t)\mathbb{E}\|\boldsymbol{\delta}_{t-1}\|$, while the second term decreases with the failure rate $1 - q_t$. $\prod_{j=t+1}^{T_d} \kappa_j$ characterizes how this error propagates through the subsequent federated updates. When $q_t = 1$, the second term vanishes and the result reduces to the ideal clean replay bound. Therefore, when the local curvature range is small, the federated update remains locally stable, and the upstream detection and correction are reliable, FedHSR can reconstruct a trajectory close to the clean training trajectory.

4 Experiment

We evaluate FedHSR in terms of overall accuracy, trajectory and response repair quality, computational cost, detection-delay robustness, and component contributions. The efficiency trade-off is reported in Appendix F.4.

Table 1: Global test accuracy (%) under client-heterogeneous label noise. The best and second-best results in each column are highlighted in bold and underlined, respectively.

	CIFAR-10						CIFAR-100					
	$\beta = 1.0$			$\beta = 0.5$			$\beta = 1.0$			$\beta = 0.5$		
Method	Sym	Pair	Mixed	Sym	Pair	Mixed	Sym	Pair	Mixed	Sym	Pair	Mixed
FedAvg	66.27	63.57	65.89	59.58	56.91	58.48	39.17	38.21	38.70	37.01	36.36	38.03
DivideMix	50.95	47.01	49.44	45.45	43.88	37.83	22.21	21.12	22.36	21.43	19.09	19.81
FedCorr	67.84	63.12	64.97	56.23	54.02	59.71	<u>53.58</u>	<u>53.96</u>	<u>53.43</u>	<u>43.92</u>	<u>48.22</u>	<u>47.19</u>
FedNoRo	<u>76.25</u>	<u>76.96</u>	<u>77.04</u>	<u>71.60</u>	<u>71.63</u>	<u>75.01</u>	41.78	40.26	41.26	40.65	39.00	40.21
FedFixer	67.10	63.70	64.03	57.12	57.53	60.64	31.15	30.66	31.33	30.16	28.99	29.05
FedDiv	57.20	56.41	58.26	48.78	48.20	51.22	32.08	34.57	33.99	31.11	31.02	31.18
FedELC	73.12	71.05	70.27	68.54	70.62	69.52	37.33	34.88	40.09	37.11	34.86	38.10
FedClean	42.96	42.99	45.03	36.68	36.10	37.68	19.84	19.41	23.77	16.47	20.17	18.46
FedRG	67.38	65.11	66.95	63.77	63.91	62.45	29.60	28.79	29.13	27.33	26.95	27.00
FedHSR	**76.97**	**77.84**	**77.83**	**72.65**	**71.76**	**75.20**	**54.27**	**55.44**	**54.59**	**46.31**	**48.93**	**49.62**

4.1 Experimental Setup

Datasets and Models. Our experimental settings follow existing federated learning with noisy-label studies Wu et al. (2023); Wen et al. (2026); Jiang et al. (2025). We evaluate FedHSR on CIFAR-10 and CIFAR-100, both of which contain 50,000 training images and 10,000 test images with a resolution of 32×32 Krizhevsky et al. (2009). We use a CIFAR-style ResNet-18 for both datasets He et al. (2016). All methods use the same training/test split, client partition, and noisy labels under each experimental condition. Dataset, preprocessing, partitioning, noise construction, and training details are provided in Appendix C. Unless stated otherwise, all clients participate in every communication round, matching the full-participation setting used in our theoretical analysis.

We construct statistically heterogeneous clients using the balanced Dirichlet partitioning adopted in prior federated noisy-label studies Wu et al. (2023); Wen et al. (2026). For each class, its samples are distributed among clients according to a Dirichlet distribution with concentration parameter β, where a smaller β indicates stronger data heterogeneity. The balanced partition further constrains the client data sizes to avoid extremely imbalanced sample quantities.

Noise Settings. We consider two representative noisy-label settings: heterogeneous client noise and uniform client noise Wu et al. (2023); Wen et al. (2026). For heterogeneous client noise, clients contain different noise ratios that increase linearly from zero toward 0.4. We evaluate two levels of data heterogeneity, $\beta \in \{1.0, 0.5\}$, together with three noise types: symmetric noise (*Sym*), pair-flip noise (*Pair*), and their client-wise mixture (*Mixed*). For uniform client noise, all clients contain noisy labels with a target noise ratio of 40%. We consider both symmetric and pair-flip noise, where the corrupted labels are sampled either from the local label support or from the complete global label space, resulting in four conditions: *Local-Sym*, *Global-Sym*, *Local-Pair*, and *Global-Pair*.

Baselines. We compare with FedAvg McMahan et al. (2017) and representative noisy-label learning methods, including DivideMix Li et al. (2020a), FedCorr Xu et al. (2022), FedNoRo Wu et al. (2023), FedFixer Ji et al. (2024), FedDiv Li et al. (2024), FedELC Jiang et al. (2024), FedClean Jiang & Zhang (2025), and FedRG Wen et al. (2026). For FedHSR, we preserve the original noise detection, local training objective, and aggregation strategy of the corresponding method, and only apply historical trajectory repair after noisy samples are identified.

Evaluation Metric. We evaluate the global model on the clean test set after each communication round and report test accuracy.

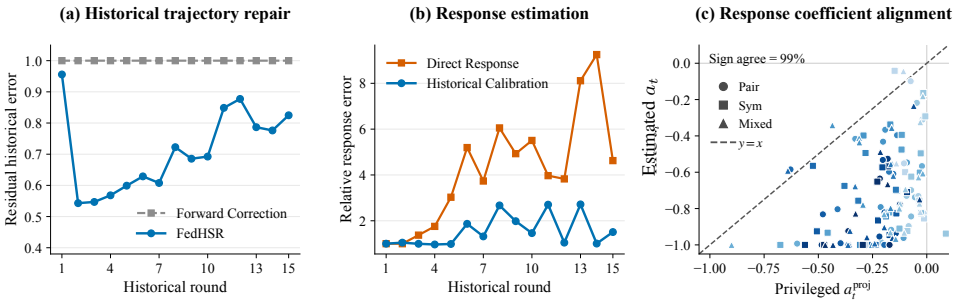


Figure 4: Historical repair diagnostics under controlled oracle detection. (a) Residual trajectory error normalized by Forward Correction. (b) Relative clean client response error. (c) Estimated and reference response coefficients across three noise types and three seeds.

4.2 Overall Effectiveness

We first examine whether FedHSR improves upon existing state of the art methods. Since FedHSR is designed to complement existing noise detectors, we compare methods based on the two dominant detection paradigms. The first paradigm detects unreliable clients when noise severity differs across clients. Table 1 reports this setting on CIFAR-10 and CIFAR-100. Across different corruption types and levels of data heterogeneity, FedHSR builds on FedNoRo for CIFAR-10 and FedCorr for CIFAR-100 and consistently achieves the strongest performance. The second paradigm detects noisy samples within each client. Table 2 evaluates this setting under uniform client noise on CIFAR-10. Since every client contains noisy samples, the advantage of client level detection diminishes, while FedRG's sample level detector provides the strongest baseline. Applying FedHSR to FedRG further improves its performance across all global and local corruption settings. These results show that FedHSR can build on the more suitable detection method in each noise scenario and provide further improvement across datasets, data heterogeneity, and corruption mechanisms.

Table 2: Global test accuracy (%) under the all-client noise setting on CIFAR-10.

Method	Global-Sym	Global-Pair	Local-Sym	Local-Pair
FedAvg	65.97	53.44	44.32	53.45
DivideMix	35.55	39.13	21.78	28.41
FedCorr	54.08	45.94	24.83	43.65
FedNoRo	63.85	47.86	64.01	51.98
FedFixer	55.85	52.91	41.11	47.79
FedDiv	40.05	46.65	37.60	36.22
FedELC	61.58	46.29	63.29	51.63
FedClean	41.21	29.06	32.35	34.27
FedRG	70.01	68.57	65.75	63.87
FedHSR	**70.60**	**70.17**	**66.14**	**64.90**

4.3 Ablation Study

In this section, we analyze the effectiveness of the components introduced by FedHSR. Our main contribution is Historical Response Calibration, which estimates how clean clients respond to the model shift caused by historical repair. We therefore construct the following variants. The Backbone performs no historical repair; Replay Only replays the detected noisy clients without estimating the clean client response; Replay + Fixed uses the same response coefficient across all rounds; and FedHSR estimates a round specific coefficient directly from the observed history. Appendix E.4 provides the complete variant definitions. Replay Only and Replay + Fixed are not consistently effective, while FedHSR outperforms the evaluated fixed response variant across all noise types. This provides empirical evidence for adapting the response coefficient to the historical trajectory.

Table 3: Controlled response calibration ablation (%).

Variant	Pair	Mixed	Sym
Backbone	76.96	76.87	76.32
Replay Only	76.69	76.64	76.17
Replay + Fixed	77.14	76.49	76.71
FedHSR	**77.84**	**77.32**	**76.90**

4.4 Historical Repair Quality

The theoretical result provides a stability bound for scalar response approximation under local curvature assumptions, but it does not by itself establish whether the coefficient inferred from historical changes agrees with the response induced by the actual repair displacement. In this section, we therefore examine both whether the trajectory reconstructed by FedHSR is close to the reference trajectory and whether the historical estimate agrees with the evaluation-only reference response. To isolate repair quality from differences among detection methods, we use a controlled oracle detection protocol. The reference trajectory and true responses are used only for evaluation and are never available to FedHSR. Appendix D.3 gives the complete protocol.

Figure 4(a) shows that FedHSR remains closer to the reference trajectory than Forward Correction, while Figure 4(b) shows that Historical Response Calibration yields a smaller response estimation error than Direct Response. Figure 4(c) compares the estimated coefficient a_t with the oracle reference coefficient a_t^{proj}. The two coefficients have the same sign in 99% of cases, and most points lie below $y = x$. This shows that the historical estimator reliably captures the direction of the true clean client response, while the observed deviation from $y = x$ indicates that this evidence should not be interpreted as exact recovery of its magnitude.

4.5 Robustness under Practical Detection Conditions

In this section, we study how delayed detection arises under partial participation and whether FedHSR remains effective under different detection delays. We evaluate four delay lengths and compare the resulting recovery behavior. Appendix E provides the complete progressive detection and detection delay protocols.

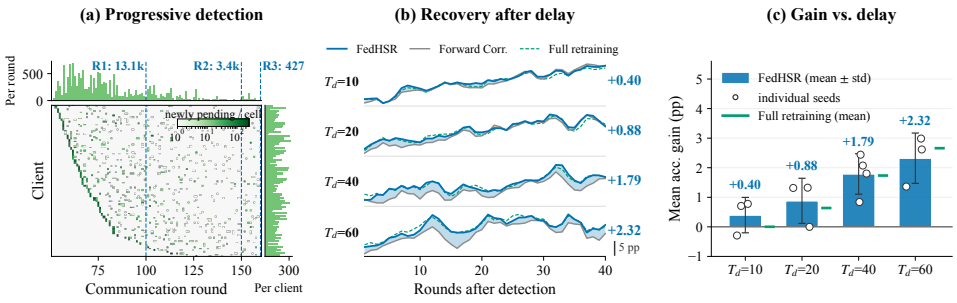


Figure 5: Progressive detection and recovery from delayed noise. (a) New detections across clients and rounds, with three repair events. (b) Recovery trajectories after four detection delays. (c) Mean accuracy gain over Forward Correction; dots denote seeds and green lines denote Full Retraining.

Figure 5(a) shows that detection events arise throughout training even under partial participation, confirming that noise can be detected progressively rather than in a single reveal. Figure 5(b) shows that FedHSR repairs the affected trajectory under every evaluated delay and remains close to Full Retraining. Figure 5(c) further shows that the benefit persists across different random seeds rather than depending on one selected run. Together, these results demonstrate that historical repair remains effective when detection is delayed under partial participation.

5 Conclusion

In this work, we formulate delayed noisy label detection in federated learning, where noisy samples affect the training trajectory before being identified. To address this problem, we propose FedHSR, which repairs historical influence through Historical Replay and estimates clean client responses through Historical Response Calibration. We bound the response estimation error and its propagation under imperfect detection and correction. Experiments across datasets, heterogeneity levels, and noise settings show consistent improvements over client and sample detection methods.

REPRODUCIBILITY STATEMENT

Section 3 presents the formulation, method, and assumptions used in the theoretical analysis. The complete repair procedure is provided in Algorithm 1 in Appendix A, and the proofs are provided in Appendix B. Section 4 presents the main experimental design and results. Appendices C–G provide the complete experimental details, including the datasets, model architectures, data partitions, noise construction, baseline implementations, repair protocols, additional results, efficiency measurements, and computing environment. All experiments use publicly available benchmark datasets and standard model architectures. The source code and configuration files are publicly available.

AI USE STATEMENT

Generative AI tools were used to assist with language polishing, readability, organization, formatting, and drafting or refining prose that describes experimental observations supplied by the authors. They were not used to generate or modify experimental data or results, run experiments, implement the proposed method, or formulate the mathematical claims and proofs. The authors independently designed the research, verified the technical content, calculations, experimental results, and citations, and reviewed all AI-assisted text. The authors take full responsibility for the content of this paper.

REFERENCES

Jonathan Barzilai and Jonathan M Borwein. Two-point step size gradient methods. *IMA journal of numerical analysis*, 8(1):141–148, 1988.

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

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE symposium on security and privacy (SP)*, pp. 141–159. IEEE, 2021.

Xiaoyu Cao, Jinyuan Jia, Zaixi Zhang, and Neil Zhenqiang Gong. Fedrecover: Recovering from poisoning attacks in federated learning using historical information. In *2023 IEEE Symposium on Security and Privacy (SP)*, pp. 1366–1383. IEEE, 2023.

Yu-Hong Dai, Yakui Huang, and Xin-Wei Liu. A family of spectral gradient methods for optimization: Y.-h. dai et al. *Computational Optimization and Applications*, 74(1):43–65, 2019.

Xiuwen Fang and Mang Ye. Robust federated learning with noisy and heterogeneous clients. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10062–10071. IEEE, 2022.

Eduard Gorbunov, Filip Hanzely, and Peter Richtárik. Local sgd: Unified theory and new efficient methods. In *International Conference on Artificial Intelligence and Statistics*, pp. 3556–3564. PMLR, 2021.

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

Xinyuan Ji, Zhaowei Zhu, Wei Xi, Olga Gadyatskaya, Zilong Song, Yong Cai, and Yang Liu. Fedfixer: Mitigating heterogeneous label noise in federated learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 12830–12838, 2024.

Xiaoqian Jiang and Jing Zhang. Fedclean: A general robust label noise correction for federated learning. In *Forty-second international conference on machine learning*, 2025.

Xuefeng Jiang, Sheng Sun, Yuwei Wang, and Min Liu. Towards federated learning against noisy labels via local self-regularization. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pp. 862–873, 2022.

Xuefeng Jiang, Sheng Sun, Jia Li, Jingjing Xue, Runhan Li, Zhiyuan Wu, Gang Xu, Yuwei Wang, and Min Liu. Tackling noisy clients in federated learning with end-to-end label correction. In *Proceedings of the 33rd ACM international conference on information and knowledge management*, pp. 1015–1026, 2024.

Xuefeng Jiang, Jia Li, Nannan Wu, Zhiyuan Wu, Xujing Li, Sheng Sun, Gang Xu, Yuwei Wang, Qi Li, and Min Liu. Fnbench: Benchmarking robust federated learning against noisy labels. *arXiv preprint arXiv:2505.06684*, 2025.

Tong Jin, Siguang Chen, and Sheng-Jun Huang. Fednlu: Robust federated learning with noisy label unlearning. *Neurocomputing*, pp. 133896, 2026.

Peter Kairouz and H Brendan McMahan. Advances and open problems in federated learning. *Foundations and trends in machine learning*, 14(1-2):1–210, 2021.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020.

Shuqi Ke, Chao Huang, and Xin Liu. Quantifying the impact of label noise on federated learning. *arXiv preprint arXiv:2211.07816*, 2022.

Sangmook Kim, Wonyoung Shin, Soohyuk Jang, Hwanjun Song, and Se-Young Yun. Fedrn: Exploiting k-reliable neighbors towards robust federated learning. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 972–981, 2022.

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

Fan Lai, Xiangfeng Zhu, Harsha V Madhyastha, and Mosharaf Chowdhury. Oort: Efficient federated learning via guided participant selection. In *15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21)*, pp. 19–35, 2021.

Jichang Li, Guanbin Li, Hui Cheng, Zicheng Liao, and Yizhou Yu. Feddiv: Collaborative noise filtering for federated learning with noisy labels. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 3118–3126, 2024.

Junnan Li, Richard Socher, and Steven C. H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020a. URL `https://openreview.net/forum?id=HJgExaVtwr`.

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

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020c. URL `https://openreview.net/forum?id=HJxNAnVtDS`.

Gaoyang Liu, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu. Federaser: Enabling efficient client-level data removal from federated learning models. In *2021 IEEE/ACM 29th international symposium on quality of service (IWQOS)*, pp. 1–10. IEEE, 2021.

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

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1944–1952, 2017.

Mónica Ribero, Haris Vikalo, and Gustavo De Veciana. Federated learning under intermittent client availability and time-varying communication constraints. *IEEE Journal of Selected Topics in Signal Processing*, 17(1):98–111, 2022.

Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger R Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N Galtier, Bennett A Landman, Klaus Maier-Hein, et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1):119, 2020.

Yichen Ruan, Xiaoxi Zhang, Shu-Che Liang, and Carlee Joe-Wong. Towards flexible device participation in federated learning. In *International conference on artificial intelligence and statistics*, pp. 3403–3411. PMLR, 2021.

Vasileios Tsouvalas, Aaqib Saeed, Tanir Ozcelebi, and Nirvana Meratnia. Labeling chaos to learning harmony: Federated learning with noisy labels. *ACM Transactions on Intelligent Systems and Technology*, 15(2):1–26, 2024.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.

Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. Learning with noisy labels revisited: A study using real-world human annotations. *arXiv preprint arXiv:2110.12088*, 2021.

Tian Wen, Zhiqin Yang, Yonggang Zhang, Xuefeng Jiang, Hao Peng, Yuwei Wang, and Bo Han. Fedrg: Unleashing the representation geometry for federated learning with noisy clients. *arXiv preprint arXiv:2603.19722*, 2026.

Nannan Wu, Li Yu, Xuefeng Jiang, Kwang-Ting Cheng, and Zengqiang Yan. Fednoro: Towards noise-robust federated learning by addressing class imbalance and label noise heterogeneity. *arXiv preprint arXiv:2305.05230*, 2023.

Jingyi Xu, Zihan Chen, Tony QS Quek, and Kai Fong Ernest Chong. Fedcorr: Multi-stage federated learning for label noise correction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10174–10183. IEEE, 2022.

Yikai Yan, Chaoyue Niu, Yucheng Ding, Zhenzhe Zheng, Shaojie Tang, Qinya Li, Fan Wu, Chengfei Lyu, Yanghe Feng, and Guihai Chen. Federated optimization under intermittent client availability. *INFORMS Journal on Computing*, 36(1):185–202, 2024.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

Seunghan Yang, Hyoungseob Park, Junyoung Byun, and Changick Kim. Robust federated learning with noisy labels. *IEEE Intelligent Systems*, 37(2):35–43, 2022.

Honglin Yuan and Tengyu Ma. Federated accelerated stochastic gradient descent. *Advances in Neural Information Processing Systems*, 33:5332–5344, 2020.

Xiaochen Zhou and Xudong Wang. Federated label-noise learning with local diversity product regularization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 17141–17149, 2024.

A Complete Repair Procedure

Algorithm 1 summarizes the complete historical repair process. Historical Replay reconstructs the updates of the affected clients. Historical Response Calibration estimates the response of the clean clients. The repaired model is propagated sequentially to the detection round.

Algorithm 1 Historical Self-Repair

Require: Historical trajectory $\{\mathbf{w}_g^{t_0-1}, \dots, \mathbf{w}_g^{T_d}\}$, detected noisy clients $\mathcal{A}$, corrected supervision
Ensure: Repaired global model $\hat{\mathbf{w}}_g^{T_d}$

$\hat{\mathbf{w}}_g^{t_0-1} \leftarrow \mathbf{w}_g^{t_0-1}$
for $t = t_0, \dots, T_d$ **do**
  for $a \in \mathcal{A}$ **do**
    $\Delta_{a,\text{noisy}}^t \leftarrow \textsc{Replay}(a, \mathbf{w}_g^{t-1})$
    $\Delta_{a,\text{clean}}^t \leftarrow \textsc{CleanReplay}(a, \hat{\mathbf{w}}_g^{t-1})$
  end for
  $\mathbf{h}_t \leftarrow (\mathbf{w}_g^t - \mathbf{w}_g^{t-1}) - \sum_{a\in\mathcal{A}} p_a^t \Delta_{a,\text{noisy}}^t$
  $\widetilde{\mathbf{w}}_g^t \leftarrow \hat{\mathbf{w}}_g^{t-1} + \mathbf{h}_t + \sum_{a\in\mathcal{A}} p_a^t \Delta_{a,\text{clean}}^t$
  $\boldsymbol{\delta}_{t-1} \leftarrow \hat{\mathbf{w}}_g^{t-1} - \mathbf{w}_g^{t-1}$
  if $t = t_0$ **then**
    $\widehat{R}_t \leftarrow \mathbf{0}$
  else
    $\mathbf{s}_t \leftarrow \mathbf{w}_g^{t-1} - \mathbf{w}_g^{t-2}$, $\mathbf{y}_t \leftarrow \mathbf{h}_t - \mathbf{h}_{t-1}$
    $a_t \leftarrow \langle \mathbf{y}_t, \mathbf{s}_t \rangle / \|\mathbf{s}_t\|_2^2$, $\widehat{R}_t \leftarrow a_t \boldsymbol{\delta}_{t-1}$
  end if
  $\hat{\mathbf{w}}_g^t \leftarrow \widetilde{\mathbf{w}}_g^t + \widehat{R}_t$
end for
return $\hat{\mathbf{w}}_g^{T_d}$

B Proofs

B.1 Proof of Proposition 1

Proposition 1 (Stability of Scalar Response Approximation). Under Assumption 1 and $\|\mathbf{w}_g^{t-1} - \mathbf{w}_g^{t-2}\|_2 > 0$, the response residual $\boldsymbol{\varepsilon}_t = R_t - \widehat{R}_t$ satisfies

$$\left\| R_t - \widehat{R}_t \right\| \le \eta(L_t - \mu_t) \left\| \boldsymbol{\delta}_{t-1} \right\|. \tag{7}$$

Proof. We first analyze the response rate a_t estimated from the historical trajectory. For compactness within this proof, let $\mathbf{s}_t = \mathbf{w}_g^{t-1} - \mathbf{w}_g^{t-2}$ and $\mathbf{y}_t = H(\mathbf{w}_g^{t-1}) - H(\mathbf{w}_g^{t-2})$. These are only shorthands for the quantities written explicitly in Eq. equation 4. Since $H(\mathbf{w}) = -\eta \nabla F(\mathbf{w})$, we have

$$\mathbf{y}_t = -\eta \left[\nabla F(\mathbf{w}_g^{t-1}) - \nabla F(\mathbf{w}_g^{t-2}) \right]. \tag{8}$$

Because $\mathbf{w}_g^{t-1} = \mathbf{w}_g^{t-2} + \mathbf{s}_t$, the fundamental theorem of calculus gives

$$\nabla F(\mathbf{w}_g^{t-1}) - \nabla F(\mathbf{w}_g^{t-2}) = \int_0^1 \nabla^2 F(\mathbf{w}_g^{t-2} + \tau \mathbf{s}_t) \mathbf{s}_t \, d\tau. \tag{9}$$

Define the average Hessian along this historical displacement as

$$\mathbf{A}_t^s = \int_0^1 \nabla^2 F(\mathbf{w}_g^{t-2} + \tau \mathbf{s}_t) \, d\tau. \tag{10}$$

Then

$$\mathbf{y}_t = -\eta \mathbf{A}_t^s \mathbf{s}_t. \tag{11}$$

Under Assumption 1, every Hessian along the local segment satisfies

$$\mu_t \mathbf{I} \preceq \nabla^2 F(\mathbf{w}) \preceq L_t \mathbf{I}.$$

Integrating this inequality along the segment preserves the Loewner order, and therefore

$$\mu_t \mathbf{I} \preceq \mathbf{A}_t^s \preceq L_t \mathbf{I}. \tag{12}$$

Substituting Eq. equation 11 into the definition of a_t gives

$$\begin{aligned} a_t &= \frac{\langle \mathbf{y}_t, \mathbf{s}_t \rangle}{\|\mathbf{s}_t\|_2^2} \\ &= -\eta \frac{(\mathbf{s}_t)^\top \mathbf{A}_t^s \mathbf{s}_t}{\|\mathbf{s}_t\|_2^2}. \end{aligned} \tag{13}$$

The term $(\mathbf{s}_t)^\top \mathbf{A}_t^s \mathbf{s}_t / \|\mathbf{s}_t\|_2^2$ is the Rayleigh quotient of $\mathbf{A}_t^s$. According to Eq. equation 12,

$$\mu_t \leq \frac{(\mathbf{s}_t)^\top \mathbf{A}_t^s \mathbf{s}_t}{\|\mathbf{s}_t\|_2^2} \leq L_t. \tag{14}$$

Hence,

$$-\eta L_t \leq a_t \leq -\eta \mu_t. \tag{15}$$

We next analyze the exact response required by the repaired trajectory. Recall that

$$\boldsymbol{\delta}_{t-1} = \hat{\mathbf{w}}_g^{t-1} - \mathbf{w}_g^{t-1},$$

and therefore

$$R_t = H(\mathbf{w}_g^{t-1} + \boldsymbol{\delta}_{t-1}) - H(\mathbf{w}_g^{t-1}).$$

Using again $H(\mathbf{w}) = -\eta \nabla F(\mathbf{w})$, we obtain

$$R_t = -\eta \left[\nabla F(\mathbf{w}_g^{t-1} + \boldsymbol{\delta}_{t-1}) - \nabla F(\mathbf{w}_g^{t-1}) \right]. \tag{16}$$

Similarly, define the average Hessian along the repair displacement as

$$\mathbf{A}_t^\delta = \int_0^1 \nabla^2 F(\mathbf{w}_g^{t-1} + \tau \boldsymbol{\delta}_{t-1}) \, d\tau. \tag{17}$$

Then

$$R_t = -\eta \mathbf{A}_t^\delta \boldsymbol{\delta}_{t-1}. \tag{18}$$

By Assumption 1,

$$\mu_t \mathbf{I} \preceq \mathbf{A}_t^\delta \preceq L_t \mathbf{I}. \tag{19}$$

Historical Response Calibration estimates this response as

$$\widehat{R}_t = a_t \boldsymbol{\delta}_{t-1}.$$

Accordingly, Eq. equation 4 defines the exact residual as $\boldsymbol{\varepsilon}_t = R_t - a_t \boldsymbol{\delta}_{t-1}$. Combining this definition with Eq. equation 18 gives

$$\begin{aligned} \|\boldsymbol{\varepsilon}_t\| &= \left\| \left(-\eta \mathbf{A}_t^\delta - a_t \mathbf{I} \right) \boldsymbol{\delta}_{t-1} \right\| \\ &\leq \left\| -\eta \mathbf{A}_t^\delta - a_t \mathbf{I} \right\|_2 \|\boldsymbol{\delta}_{t-1}\|. \end{aligned} \tag{20}$$

From Eq. equation 19, all eigenvalues of $-\eta \mathbf{A}_t^\delta$ lie in $[-\eta L_t, -\eta \mu_t]$. According to Eq. equation 15, a_t lies in exactly the same interval. Therefore, the maximum distance between a_t and any eigenvalue of $-\eta \mathbf{A}_t^\delta$ is at most the width of this interval:

$$\left\| -\eta \mathbf{A}_t^\delta - a_t \mathbf{I} \right\|_2 \leq \eta (L_t - \mu_t). \tag{21}$$

Substituting this result into Eq. equation 20, we finally obtain

$$\|\boldsymbol{\varepsilon}_t\| \leq \eta (L_t - \mu_t) \|\boldsymbol{\delta}_{t-1}\|, \tag{22}$$

which completes the proof. □

B.2 Proof of Theorem 1

Theorem 1 (Historical Trajectory Repair). Suppose the repaired trajectory and the clean trajectory share the same global model before the noisy sample first affects training, i.e., $\hat{\mathbf{w}}_g^{t_0-1} = \bar{\mathbf{w}}_g^{t_0-1}$. Under Assumption 1 and the local Lipschitz condition on $\Phi_t(\cdot)$, suppose each client level detection and repair decision at round t succeeds with probability at least q_t, and a failed decision on client k changes the aggregate transition by at most $p_k^t B_{k,t}$. Then the reconstructed global model at the detection round T_d satisfies

$$\mathbb{E}\left\|\hat{\mathbf{w}}_g^{T_d} - \bar{\mathbf{w}}_g^{T_d}\right\| \leq \sum_{t=t_0}^{T_d} \left(\prod_{j=t+1}^{T_d} \kappa_j \right) \left[\eta(L_t - \mu_t)\mathbb{E}\left\|\boldsymbol{\delta}_{t-1}\right\| + (1-q_t)\sum_{k=1}^{K} p_k^t B_{k,t} \right]. \tag{23}$$

Proof. Let $\Phi_t(\mathbf{w})$ denote the clean federated transition at round t. The clean trajectory therefore satisfies

$$\bar{\mathbf{w}}_g^t = \Phi_t(\bar{\mathbf{w}}_g^{t-1}). \tag{24}$$

For the repaired trajectory, Historical Replay reconstructs the clean contribution of the noisy client from the current repaired model $\hat{\mathbf{w}}_g^{t-1}$. If the exact clean-client response R_t were available, the same clean transition would also be applied to $\hat{\mathbf{w}}_g^{t-1}$. However, Historical Response Calibration uses the estimated response $\widehat{R}_t = R_t - \boldsymbol{\varepsilon}_t$ instead. Let $\boldsymbol{\xi}_t$ denote the net transition error caused by unsuccessful detection or correction. Hence, the repaired model can be written as

$$\hat{\mathbf{w}}_g^t = \Phi_t(\hat{\mathbf{w}}_g^{t-1}) - \boldsymbol{\varepsilon}_t + \boldsymbol{\xi}_t. \tag{25}$$

Importantly, Eq. equation 25 does not assume that the current repaired model $\hat{\mathbf{w}}_g^{t-1}$ already coincides with the clean model $\bar{\mathbf{w}}_g^{t-1}$. Any reconstruction error accumulated in previous rounds is retained in the difference between these two starting models.

Define the trajectory reconstruction error at round t as

$$E_t = \left\|\hat{\mathbf{w}}_g^t - \bar{\mathbf{w}}_g^t\right\|. \tag{26}$$

Combining Eqs. equation 24 and equation 25, we obtain

$$\begin{aligned} E_t &= \left\|\Phi_t(\hat{\mathbf{w}}_g^{t-1}) - \Phi_t(\bar{\mathbf{w}}_g^{t-1}) - \boldsymbol{\varepsilon}_t + \boldsymbol{\xi}_t\right\| \\ &\leq \left\|\Phi_t(\hat{\mathbf{w}}_g^{t-1}) - \Phi_t(\bar{\mathbf{w}}_g^{t-1})\right\| + \left\|\boldsymbol{\varepsilon}_t\right\| + \left\|\boldsymbol{\xi}_t\right\|. \end{aligned} \tag{27}$$

Since $\Phi_t(\cdot)$ is locally Lipschitz with constant κ_t,

$$\begin{aligned} \left\|\Phi_t(\hat{\mathbf{w}}_g^{t-1}) - \Phi_t(\bar{\mathbf{w}}_g^{t-1})\right\| &\leq \kappa_t \left\|\hat{\mathbf{w}}_g^{t-1} - \bar{\mathbf{w}}_g^{t-1}\right\| \\ &= \kappa_t E_{t-1}. \end{aligned} \tag{28}$$

Therefore,

$$E_t \leq \kappa_t E_{t-1} + \left\|\boldsymbol{\varepsilon}_t\right\| + \left\|\boldsymbol{\xi}_t\right\|. \tag{29}$$

The first term in Eq. equation 29 represents the propagation of the reconstruction error accumulated before round t. The second term represents the new approximation error introduced by Historical Response Calibration at round t, and the third term represents the error introduced by unsuccessful detection or correction.

According to Proposition 1,

$$\left\|\boldsymbol{\varepsilon}_t\right\| \leq \eta(L_t - \mu_t)\left\|\boldsymbol{\delta}_{t-1}\right\|. \tag{30}$$

By the assumed detection-and-repair success rate and bounded effect of each failed decision, the triangle inequality and linearity of expectation give

$$\mathbb{E}\left\|\boldsymbol{\xi}_t\right\| \leq (1-q_t)\sum_{k=1}^{K} p_k^t B_{k,t}. \tag{31}$$

No independence among client decisions is required for this inequality. Substituting both bounds into Eq. equation 29 and taking expectations gives

$$\mathbb{E}E_t \le \kappa_t \mathbb{E}E_{t-1} + \eta(L_t - \mu_t)\mathbb{E}\left\|\boldsymbol{\delta}_{t-1}\right\| + (1-q_t)\sum_{k=1}^{K} p_k^t B_{k,t}. \tag{32}$$

Before round t_0, the noisy sample has not affected federated training. Therefore, the original, repaired, and clean trajectories share the same global model, and

$$E_{t_0-1} = \left\|\hat{\mathbf{w}}_g^{t_0-1} - \bar{\mathbf{w}}_g^{t_0-1}\right\| = 0. \tag{33}$$

Applying Eq. equation 32 recursively until the detection round T_d yields

$$\mathbb{E}E_{T_d} \le \sum_{t=t_0}^{T_d} \left(\prod_{j=t+1}^{T_d} \kappa_j \right) \left[\eta(L_t - \mu_t)\mathbb{E}\left\|\boldsymbol{\delta}_{t-1}\right\| + (1-q_t)\sum_{k=1}^{K} p_k^t B_{k,t} \right]. \tag{34}$$

Since $E_{T_d} = \|\hat{\mathbf{w}}_g^{T_d} - \bar{\mathbf{w}}_g^{T_d}\|$, we obtain Eq. equation 23, which completes the proof. □

C Experimental Settings

This appendix first describes the common experimental setup and baselines. It then separates the repair implementation, additional experiment settings, additional results, and deployment details.

C.1 Datasets and Model Architecture

We evaluate all methods on CIFAR-10 and CIFAR-100 Krizhevsky et al. (2009). Both datasets contain 50,000 training images and 10,000 test images of size 32×32. CIFAR-10 contains 10 classes, whereas CIFAR-100 contains 100 classes. We use the official training and test splits and evaluate the global model on the clean test set after every communication round. All experiments use a CIFAR-style ResNet-18 He et al. (2016), whose first convolution operates directly on 32×32 inputs and does not use the ImageNet stem or initial max-pooling layer.

For weak training augmentation, we apply random horizontal flipping, a random 32×32 crop with reflection padding of four pixels, conversion to a tensor, and channel-wise normalization. For CIFAR-10, the normalization mean and standard deviation are $(0.4914, 0.4822, 0.4465)$ and $(0.2470, 0.2435, 0.2616)$, respectively. For CIFAR-100, they are $(0.5071, 0.4867, 0.4408)$ and $(0.2675, 0.2565, 0.2761)$, respectively. FedRG additionally constructs a strong view by applying RandAugment with three operations and magnitude five before the weak augmentation pipeline.

C.2 Baseline Methods

All baselines are reimplemented or ported into the same federated training framework. The framework standardizes data loading, model wrappers, logging, and checkpointing while preserving each method's detector, local objective, and aggregation rule.

FedAvg. FedAvg McMahan et al. (2017) optimizes cross entropy on local noisy data and averages client updates according to local data sizes without explicit noise handling.

DivideMix. DivideMix Li et al. (2020a) is applied independently on each participating client. Two global networks fit Gaussian mixture models to per sample losses, divide local data into labeled and unlabeled subsets, and provide targets for one another during local MixMatch training. The networks are aggregated separately with FedAvg, and the first is used for evaluation.

FedCorr. FedCorr Xu et al. (2022) uses multiple stages to identify noisy clients and samples before label correction. We retain its detection, correction, local training, and aggregation rules. It provides the detections used by FedHSR in the CIFAR-100 experiments with heterogeneous client noise.

FedNoRo. FedNoRo Wu et al. (2023) detects noisy clients from per client and per class loss statistics with a Gaussian mixture model. Its robust stage combines dedicated local objectives, knowledge distillation, and noise aware aggregation. It provides the detections used by FedHSR in the CIFAR-10 experiments with heterogeneous client noise.

FedFixer. FedFixer Ji et al. (2024) combines personalized and global knowledge to identify reliable samples and mitigate heterogeneous label noise. We retain its sample selection and model training procedure.

FedDiv. FedDiv Li et al. (2024) constructs a collaborative noise filter across clients and uses the filtered samples for robust federated optimization. Its filter and local objective are preserved.

FedELC. FedELC Jiang et al. (2024) estimates client noise levels and corrects labels for clients identified as noisy. We preserve its client detection, correction objective, and aggregation procedure.

FedClean. FedClean Jiang & Zhang (2025) combines sample selection, label correction, and robust aggregation. These components are retained in our implementation.

FedRG. FedRG Wen et al. (2026) first learns representations through a SimCLR warm-up and then uses representation geometry with vMF/GMM modeling to infer a sample level clean or noisy mask. Its second stage jointly optimizes the classifier and personalized transition matrices. It provides the detections used by FedHSR in the experiments where every client is noisy.

C.3 Federated Data Partitioning

The FNBench/FedNoRo protocol Jiang et al. (2025) contains 100 clients, while the FedRG protocol contains 10 clients. We use a capacity-constrained Dirichlet partition to construct statistically heterogeneous client data. For each class, client proportions are sampled from a Dirichlet distribution with concentration $\beta \in \{0.5, 1.0\}$. Once a client reaches the average data size N/K, its probability of receiving additional samples from subsequent classes is set to zero. The allocation is resampled until every client contains at least 10 training samples. This procedure limits clients from substantially exceeding the average data size while preserving heterogeneous class proportions; it does not force all clients to contain exactly the same number of samples.

C.4 Noise Construction

Client-heterogeneous noise. For the symmetric and pair-flip settings, the noise rate of client i is assigned deterministically as

$$\rho_i = \rho_{\min} + \frac{i}{K}(\rho_{\max} - \rho_{\min}), \qquad i = 0, \ldots, K-1, \tag{35}$$

where $\rho_{\min} = 0$ and $\rho_{\max} = 0.4$. Consequently, the implemented range is $[0, 0.4)$ rather than including the upper endpoint. For example, with 10 clients, the rates are $[0, 0.04, 0.08, \ldots, 0.36]$. Accordingly, "client-heterogeneous" refers to heterogeneous client-level noise rates and the use of a client-level detector; it does not mean that only a strict subset of clients must have nonzero noise. This differs from the all-client protocol below, in which every client is assigned the same target rate and noise is detected at sample level.

The Mixed setting assigns a noise mechanism at the client level. The first half of the clients use symmetric noise and the second half use pair-flip noise. Within each half, the noise rates are independently ramped from zero toward 0.4 using the same rule above, with the size of that half replacing K. Thus, a client contains one corruption mechanism rather than a sample-level mixture of symmetric and pair-flip noise.

In the FNBench protocol, global pair-flip noise maps class y to $(y + 1) \bmod C$. Symmetric noise replaces a selected label uniformly with a different class.

All-client noise. In the FedRG protocol, every client and every locally present class has a target noise rate of 0.4. For client k and class c, we corrupt $\lfloor 0.4 n_{k,c} \rfloor$ samples, where $n_{k,c}$ is the number

of local samples from that class. The realized client-level rate can therefore be slightly below 0.4 because of per-class rounding.

Localized corruption draws incorrect labels only from the set of classes present on the corresponding client. Local symmetric noise samples uniformly from the other locally present classes, while local pair-flip noise sorts the local class set and performs a cyclic mapping within that set. If a client contains only one class, localized corruption cannot produce a different label and the selected label remains unchanged. Globalized corruption instead uses all C classes. In the FedRG experiments, global and local pair-flip noise use the previous class in the corresponding cyclic ordering, i.e., the global mapping is $y \mapsto (y-1) \bmod C$.

C.5 Training Configuration

The main experiments run for 160 communication rounds. Unless stated otherwise, all clients participate in every communication round, matching the fixed full-participation operator used in the theoretical response analysis. The progressive-detection and detection-delay studies are separate empirical stress tests with partial participation and are described below. We preserve each baseline's detector, local objective, method-specific aggregation, and training stages when porting it into our common federated learning framework. In particular, methods with a dedicated detection or warm-up stage retain that protocol. Each backbone and its FedHSR-augmented counterpart use matching training and detection settings.

FedNoRo uses a time-varying coefficient for its knowledge-distillation loss, which is distinct from learning-rate scheduling. Between the beginning and end of its correction stage, the coefficient is

$$\lambda_{\mathrm{KD}}(t) = 0.8 \exp\left[-5\left(1 - \frac{t - T_b}{T_e - T_b}\right)^2\right]. \tag{36}$$

C.6 Evaluation and Reporting Protocol

We evaluate the clean global test accuracy after each communication round. Each backbone and its FedHSR counterpart use matching data partitions, noise realizations, and training protocols.

D Repair Implementation and Quality Protocol

D.1 Historical Replay

The server stores historical global checkpoints, aggregation weights, and the random-number generator state immediately before local training. It does not store every historical client model or optimizer. When repair is triggered, a detected noisy client is rerun on the original historical global checkpoint, original labels, and restored random state to reproduce its factual noisy update. The same client is then trained from the repaired checkpoint using the corrected training rule to obtain its repaired update. Consequently, storage grows with the number of historical global checkpoints rather than with the product of clients and communication rounds.

The clean-client aggregate at round t is recovered without rerunning clean clients. Let $G_t = \mathbf{w}_g^t - \mathbf{w}_g^{t-1}$ denote the observed global increment and $\mathcal{N}_t$ the detected noisy clients that participated in that round. We compute

$$H_t = G_t - \sum_{i \in \mathcal{N}_t} p_i^t \Delta_i^t, \tag{37}$$

where each Δ_i^t is reproduced using the saved historical state.

Here, "reproduced" refers to the factual update generated in the original run, rather than the corrected counterfactual update. Conditional on the saved global checkpoint, local data and labels, aggregation weight, round-dependent training rule, and restored random state, the common FNBench/FedNoRo local routine is deterministic up to floating-point nondeterminism. Replay fidelity can be audited

using

$$e_{i,t}^{\text{replay}} = \frac{\left\| \Delta_{i,\text{replay}}^{t} - \Delta_{i,\text{original}}^{t} \right\|_2}{\left\| \Delta_{i,\text{original}}^{t} \right\|_2 + \epsilon}. \tag{38}$$

For a backbone with persistent client-specific state, that state is also part of the replay state required by this definition; a global checkpoint and RNG state alone are not asserted to be sufficient.

Reconstruction of FedRG state. FedRG does not store a separate historical transition matrix or sample mask for every round. At the beginning of Stage II, the implementation records the initial global model, the Python, NumPy, PyTorch, and CUDA random states, and the client-sampler state. Historical repair restarts from this Stage-II state and follows the original client order and random stream. For each client, its personalized transition matrix T, vMF state, and concentration parameter κ are initialized once at the Stage-II starting point and evolve across that client's subsequent replayed rounds; they are therefore reconstructed along the historical trajectory rather than reinitialized independently in each round.

The historical sample mask is also not stored. During replay, FedRG reruns its native detector using the reconstructed model, T, vMF state, κ, and local data. For samples revealed by delayed detection, their identifiers are forced into the noisy mask in every affected pre-detection round:

$$\widetilde{\mathcal{M}}_t = \mathcal{M}_t^{\text{FedRG}} \cup \mathcal{M}_t^{\text{delayed}}. \tag{39}$$

They are then retrained with FedRG's transition-matrix noisy loss rather than with ground-truth labels. After repair, the estimated branch replaces T, vMF, and κ only for affected clients, whereas the oracle full-retraining branch replaces these states for every client it retrains. FedRG creates a fresh local optimizer at every client invocation, so no optimizer state is carried across rounds or needs to be saved.

Replay fidelity. We separately test whether restoring the complete historical state reproduces the factual training prefix before any correction is applied. The experiment obtains zero endpoint discrepancy for both evaluated backbones. Its protocol and complete result are reported in Appendix F.1.

D.2 Response Coefficient

In implementation, the response coefficient uses the numerically stabilized form

$$a_t = \frac{\langle \mathbf{y}_t, \mathbf{s}_t \rangle}{\|\mathbf{s}_t\|_2^2 + \epsilon}, \qquad \epsilon = 10^{-12}. \tag{40}$$

At the first reconstructed round, we set $\widehat{\mathbf{R}}_{t_0} = \mathbf{0}$ because no previous observation is available.

D.3 Historical Repair Quality Protocol

Purpose and setting. To evaluate historical repair independently of detection errors, we use a controlled oracle detection protocol on CIFAR-10. We construct 100 clients with the balanced Dirichlet partition ($\beta = 1.0$) and the FNBench Pair-noise setting. This controlled diagnostic designates 50 clients as noisy and assigns their noise rates from zero toward 0.4; it is distinct from the main FNBench ramp over the full client population. All 100 clients participate in every round, and the model is a CIFAR-style ResNet-18.

Repair and reference trajectory. At round 15, the oracle detector reveals the 50 noisy clients designated for this experiment. Starting from the same initialization, FedHSR then reconstructs the historical trajectory round by round. Detected noisy clients are retrained using the FedNoRo correction objective, whereas clean clients are not retrained; Historical Response Calibration estimates the change in their aggregate update caused by the repaired starting point.

For evaluation only, we construct the exact surrogate reference trajectory $\mathbf{w}_g^{t,\text{ref}}$ induced by the FedNoRo objective used in replay. At each historical round, all clients are retrained from the counterfactual starting point using the same random state and aggregation weights as in the original

run. Neither this evaluation-only reference trajectory nor the resulting true clean-client response is available to FedHSR.

Specifically, detected noisy clients use the same FedNoRo correction objective used by Historical Replay, while unaffected clients use their original training objective. Thus, this reference is not a separate trajectory trained with ground-truth labels. Historical aggregation weights are held fixed rather than recomputed from the counterfactual model, so the diagnostic isolates update changes caused by the repaired starting point.

Evaluation metrics. For visualization, we normalize the residual trajectory error of method m by the residual error of Forward Correction at the same round:

$$D_t^{(m)} = \frac{\left\|\mathbf{w}_{g,m}^t - \mathbf{w}_g^{t,\mathrm{ref}}\right\|_2}{\left\|\mathbf{w}_{g,\mathrm{Forward}}^t - \mathbf{w}_g^{t,\mathrm{ref}}\right\|_2}, \tag{41}$$

where m is Forward Correction or FedHSR. Consequently, Forward Correction is identically one in Figure 4; values below one indicate that a method removes a larger fraction of the residual historical influence.

We additionally evaluate the clean-client response estimate. Let

$$\mathbf{R}_t = \mathbf{H}_t^{\mathrm{ref}} - \mathbf{H}_t, \tag{42}$$

where $\mathbf{H}_t$ is the aggregate update of clean clients from the original historical starting point and $\mathbf{H}_t^{\mathrm{ref}}$ is their true aggregate update from the repaired starting point. FedHSR estimates this response as $\widehat{\mathbf{R}}_t = a_t \boldsymbol{\delta}_{t-1}$, and we report

$$E_t = \frac{\left\|\widehat{\mathbf{R}}_t - \mathbf{R}_t\right\|_2}{\left\|\mathbf{R}_t\right\|_2}. \tag{43}$$

Direct Response instead sets $\widehat{\mathbf{R}}_t = \boldsymbol{\delta}_{t-1}$. A lower relative error indicates a more accurate estimate of the clean-client response.

For the coefficient diagnostic in Figure 4(c), we use the evaluation-only reference response to compute

$$a_t^{\mathrm{proj}} = \frac{\langle \mathbf{R}_t, \boldsymbol{\delta}_{t-1} \rangle}{\|\boldsymbol{\delta}_{t-1}\|_2^2 + \epsilon}. \tag{44}$$

This reference coefficient is unavailable to FedHSR and is used only to assess the direction of its historical estimate.

E Additional Experimental Settings

E.1 Repair Schedules

For FedNoRo, noisy clients are detected once at round 15. FedHSR starts from the common initialization, reconstructs rounds 0–14, and performs no further repair. For FedRG, the native vMF/GMM detector runs in every round. Newly detected samples enter a pending set, which is processed every 50 communication rounds and once more at the end of training if events remain. Each repair reconstructs the complete factual prefix from the common Stage II initialization with the cumulative detected set. In the delay experiment, the correct labels are revealed at T_d, and historical repair is triggered immediately.

E.2 Detection-Delay Protocol

The detection-delay experiment is conducted on CIFAR-10 with 100 clients and a balanced Dirichlet partition with $\beta = 0.5$. It uses the FNBench client-level noise ramp from zero toward 0.4 and globalized Pair noise with the next-class mapping $y \mapsto (y+1) \bmod C$. The backbone is ResNet-18. Ten clients are sampled per round. This experiment therefore serves as an additional partial-participation stress test outside the fixed-operator conditions of the theoretical bound.

Before the designated detection round T_d, corrupted labels participate in training normally. At T_d, an oracle sample-level detector reveals the correct labels of all corrupted samples at once. Every method then receives a fixed 40-round recovery budget. Thus, $T_d \in \{10, 20, 40, 60\}$ corresponds respectively to total training lengths of $\{50, 60, 80, 100\}$ rounds and evaluation windows 10–49, 20–59, 40–79, and 60–99. Methods within a row have identical detection time and recovery budget; results across rows have different total training lengths and are not used to claim a monotonic effect of delay.

E.3 Partial Participation with Progressive Detection

This experiment combines partial participation with the actual progressive detector used by FedRG. We train on CIFAR-10 for 160 communication rounds with 100 clients, sampling 10 clients uniformly in each round. Data are partitioned as in the uniform-client main experiment, and every client receives 40% Global-Pair label noise. Results are averaged over three random seeds.

FedRG's native vMF/GMM detector runs only on clients selected in the current round; neither inactive clients nor their samples are inspected. Newly detected samples are accumulated in a pending set. FedHSR processes all pending events every 50 communication rounds and performs a final repair at the end of training if events remain. FedRG, Replay Only, and FedHSR use identical client sampling sequences, data partitions, and noise realizations within each seed. Replay Only reconstructs detected affected client updates without Historical Response Calibration, whereas FedHSR additionally estimates the aggregate response of the other clients.

E.4 Controlled Ablation

The ablation in Table 3 isolates the contribution of Historical Response Calibration from that of noisy-client replay. It follows the CIFAR-10 client-heterogeneous setup with $\beta = 1.0$ and evaluates Pair, Sym, and Mixed noise. All repair variants start from the same initialization and reuse the same client sequence, randomness, noisy client correction objective, and aggregation weights. They can therefore be written under the common update

$$\hat{\mathbf{w}}_t = \hat{\mathbf{w}}_{t-1} + \mathbf{H}_t + \mathbf{N}_t(\hat{\mathbf{w}}_{t-1}) + \widehat{\mathbf{R}}_t, \tag{45}$$

where $\mathbf{H}_t$ denotes the factual contribution of the other clients and $\mathbf{N}_t(\hat{\mathbf{w}}_t)$ is the replayed contribution of detected noisy clients. The variants modify only $\widehat{\mathbf{R}}_t$:

- **Backbone**: FedNoRo without historical repair.
- **Replay Only**: noisy-client replay with $\widehat{\mathbf{R}}_t = \mathbf{0}$.
- **Replay + Fixed**: replay with the fixed response $\widehat{\mathbf{R}}_t = -0.1\boldsymbol{\delta}_{t-1}$.
- **FedHSR**: replay with the history dependent Historical Response Calibration estimator in Eq. equation 4.

F Additional Experimental Results

F.1 Replay Fidelity

This experiment tests whether complete factual replay reproduces the original training prefix before any correction is applied. We restore the initial model, original labels, client sequence, random stream, aggregation weights, and persistent client state. For each reconstructed prefix, we measure

$$e^{\text{end}} = \frac{\|\mathbf{w}^{\text{end}}_{\text{replay}} - \mathbf{w}^{\text{end}}_{\text{factual}}\|_2}{\|\mathbf{w}^{\text{end}}_{\text{factual}}\|_2 + \epsilon}. \tag{46}$$

Across 25 endpoints from the complete FedRG path, the mean, median, and maximum discrepancy are all zero. The audited FedNoRo prefix also has zero discrepancy. Consequently, subtracting the reproduced affected updates from the factual global increment recovers the same H_t as direct aggregation of the other factual client updates.

F.2 Progressive Detection Results

Approximately 6,300–6,600 corrupted samples in each run are first detected in later communication rounds, confirming that detection events arise progressively rather than through an oracle reveal. FedHSR improves on FedRG in all three runs.

Table 4: Test accuracy (%) with partial participation and FedRG's native progressive detector under Global Pair noise.

Seed	FedRG	FedHSR	Improvement
0	55.29	**56.61**	+1.32
1	51.19	**51.49**	+0.30
2	53.32	**54.11**	+0.79
Mean ± std	53.27 ± 2.05	$\mathbf{54.07 \pm 2.56}$	+0.80

Table 5 additionally reports the aggregate comparison with Replay Only. This variant reconstructs affected client updates without estimating the response of the remaining clients.

Table 5: Aggregate test accuracy (%) under partial participation and FedRG's native progressive detector.

Method	Test accuracy
FedRG	53.27 ± 2.05
Replay Only	$\underline{53.59 \pm 3.29}$
FedHSR	$\mathbf{54.07 \pm 2.56}$

F.3 Detection Delay Results

Table 6 reports the test accuracy for each detection delay. Since different rows begin recovery at different stages of training, the absolute accuracies are compared within each row. The delay trend in Figure 5 instead uses the mean gain over the matched Forward Correction trajectory.

Table 6: Test accuracy (%) under different detection delays.

Detection delay	Forward Correction	Direct Repair	FedHSR
10	39.63	42.29	**42.45**
20	45.83	47.13	**47.23**
40	47.43	46.34	**49.38**
60	49.04	51.67	**52.30**

F.4 Efficiency Trade-off

To characterize the efficiency trade-off, we measure only the historical repair stage while varying the total number of clients, the fraction of affected clients that must be replayed, and the detection delay. Normal backbone training is excluded from both timings. Full Retraining replays every client in the historical prefix, whereas FedHSR replays affected clients and estimates the aggregate clean-client response.

The results demonstrate a dynamic efficiency trade-off. Replaying a smaller fraction of affected clients yields a larger repair-stage speedup, while a broader repair scope reduces the efficiency advantage because more client computation must be replayed.

G Deployment and Computing Environment

Historical Replay does not require the server to cache local data or a client model for every historical round. At repair time, an affected client can rerun its existing correction routine, or an authorized

Table 7: Repair-stage scaling. Runtime excludes normal backbone training.

Clients K	Affected ratio	Delay	Full Retraining	FedHSR	Speedup
20	10%	15	222.1 s	127.1 s	1.75×
50	10%	15	242.0 s	145.7 s	1.66×
100	10%	5	83.6 s	50.2 s	1.66×
100	10%	10	168.6 s	104.3 s	1.62×
100	10%	15	259.0 s	150.4 s	1.72×
100	30%	15	263.6 s	180.6 s	1.46×
100	50%	15	270.3 s	213.0 s	1.27×

replay service can perform the same computation when it has access to the required historical data and state. If neither is available, repair for that client is deferred. Historical checkpoints and reconstructed state reside on the coordinating server and add no client storage requirement.

Experiments are implemented with PyTorch 2.6.0 and torchvision 0.21.0, using CUDA 12.4 and cuDNN 9.1.0. The server contains eight NVIDIA Tesla T4 GPUs with approximately 15 GB memory per GPU, two Intel Xeon Gold 5218R CPUs (40 physical and 80 logical cores in total), and 376.5 GiB system memory, and runs Ubuntu Linux 5.15 with glibc 2.35. For efficiency measurements, each noise setting is assigned exclusively to one T4 GPU, methods are executed sequentially on that GPU, and each process is restricted to two PyTorch CPU threads. The source code and configuration files are publicly available.