Low-Cost Privacy-Preserving Decentralized Learning

Sayan Biswas
EPFL, Switzerland

Davide Frey
Univ Rennes, Inria, CNRS, IRISA, France

Romaric Gaudel
Univ Rennes, Inria, CNRS, IRISA, France

Anne-Marie Kermarrec
EPFL, Switzerland

Dimitri Lerévérend*
Univ Rennes, Inria, CNRS, IRISA, France

Rafael Pires
EPFL, Switzerland

Rishi Sharma
EPFL, Switzerland

François Taïani
Univ Rennes, Inria, CNRS, IRISA, France

Abstract

Decentralized learning (DL) is an emerging paradigm of collaborative machine learning that enables nodes in a network to train models collectively without sharing their raw data or relying on a central server. This paper introduces Zip-DL, a privacy-aware DL algorithm that leverages correlated noise to achieve robust privacy against local adversaries while ensuring efficient convergence at low communication costs. By progressively neutralizing the noise added during distributed averaging, Zip-DL combines strong privacy guarantees with high model accuracy. Its design requires only one communication round per gradient descent iteration, significantly reducing communication overhead compared to competitors. We establish theoretical bounds on both convergence speed and privacy guarantees. Moreover, extensive experiments demonstrating Zip-DL's practical applicability make it outperform state-of-the-art methods in the accuracy vs. vulnerability trade-off. Specifically, Zip-DL *(i)* reduces membership-inference attack success rates by up to 35% compared to baseline DL, *(ii)* decreases attack efficacy by up to 13% compared to competitors offering similar utility, and *(iii)* achieves up to 59% higher accuracy to completely nullify a basic attack scenario, compared to a state-of-the-art privacy-preserving approach under the same threat model. These results position Zip-DL as a practical and efficient solution for privacy-preserving decentralized learning in real-world applications.

Keywords

decentralized learning, differential privacy, correlated noises

1 Introduction

Decentralized learning (DL) allows a collection of devices to train a global model collaboratively without sharing raw training data. This approach has drawn increasing attention from both academia [3] and industry, showcasing its potential across various sectors, including healthcare [28, 39] and autonomous vehicles [8]. In DL, each device (henceforth *node*) *(i)* trains a local model using its own data; *(ii)* exchanges this model with those of its neighbors according to the underlying communication topology; and *(iii)* averages its current local model with the models received from neighbors. This iterative process repeats until convergence is reached [27, 34]. Although training data never leaves participating nodes in DL, the models that nodes exchange still leak information. Exploiting these leaks, an honest-but-curious attacker can mount privacy attacks against participants to reveal sensitive attributes of their data. For instance, an attacker can mount a Membership-Inference Attack (MIA) [7, 37] that can reveal whether a particular sample belongs to the training set of a node.

Differential Privacy (DP) [12] is a widely-used measure of formal privacy guarantees that has been applied to the design of privacy-preserving DL [35]. DP strategically adds noise to data so that the inclusion or exclusion of a data point becomes much harder to detect. However, DP typically assumes a worst-case threat model in which an attacker can access all messages transiting on the network. As a result, although it provides robust privacy guarantees, DP tends to require high noise levels that disrupt the learning process and severely impair the system's utility.

Following existing literature [9, 14], we assume a representative threat model in which local honest-but-curious attackers can only observe the messages they receive. An attack is furthermore considered successful only if the obtained information can be linked to its contributing participant. This model covers a wide range of scenarios in which network communication is protected. Still, nodes participating in the distributed learning process can exploit their partial knowledge of the system to breach the privacy of other participants. To specifically address this threat model, Muffliato [9] introduces *Pairwise Network Differential Privacy (PNDP)*. In contrast to DP that captures a global privacy measure, PNDP tracks privacy loss at a finer level, between pairs of nodes. As a result, PNDP lends itself to lower noise levels, faster convergence, and better accuracy. Unfortunately, its use so far requires multiple rounds of averaging [9], leading to high network costs.

This paper explores the use of correlated noise to achieve PNDP without significant network costs. Correlated noise—a natural evolution of noise-based privacy methods—protects individual node inputs while minimizing the impact on model accuracy. Although systems using correlated noise show promising convergence [14], their

*Corresponding author: first.last@inria.fr

Proceedings on Privacy Enhancing Technologies 2025(3), 451–474

https://doi.org/10.56553/popets-2025-0108

Table 1: Position of our work compared to previous approaches.

Approach	Masking (RSS-NB) [14]	RSS-LB [14]	Muffliato [9]	**Zip-DL (ours)**
Formal privacy guarantees	✓	✗	✓	✓
No P2P coordination	✗	✓	✓	✓
One averaging round	✓	✓	✗	✓
Communication cost	Moderate	Low	High	Low
Impact on Convergence Rate	None	High	High	Low

privacy implications remain underexplored. Several approaches using correlated noise have been formulated [2, 19, 35], but most of them either rely on a trusted aggregator to cancel out the noises [19, 35] or on pairwise coordination between nodes, which comes at a cost either in communication or in utility [2].

We introduce Zip-DL (*Zero-summing Interference for Privacy-preserving Decentralized Learning*), a privacy-preserving algorithm that leverages correlated noise in a single communication round while offering formal privacy guarantees. To the best of our knowledge, Zip-DL (see Table 1) is the only approach *(i)* with formal guarantees that *(ii)* requires no prior pairwise coordination between nodes, and *(iii)* only requires a single averaging round per gradient step. In addition to Zip-DL, we make the following contributions:

- We prove that our approach converges while relying on a single communication round per gradient step. This powerful property results from the fact that the sum of the noise added to the communication round is zero. Moreover, our analysis shows that the impact of the noise on the convergence rate is negligible compared to state-of-the-art methods.
- We provide a formal privacy guarantee of our approach in terms of Pairwise Network Differential Privacy (PNDP).
- We conduct an extensive evaluation study comparing Zip-DL with both a state-of-the-art baseline and standard DL under threshold-based membership inference attacks (MIA) on both the CIFAR-10 and MovieLens datasets. Our results show that Zip-DL provides the best trade-off between accuracy and privacy while maintaining low communication overhead. In particular, Zip-DL reduces the success rate of MIA by up to 26 percentage points while only entailing a loss of 11 percentage points in test accuracy against baseline DL. Zip-DL also improves test accuracy by up to 59% w.r.t. to the state-of-the-art privacy-preserving baseline of Muffliato when configured to completely nullify a baseline threshold-based attack scenario.

The paper is organized as follows: Section 2 provides the necessary background and threat model. Section 3 presents the design of Zip-DL and its core properties. Sections 4 and 5 present the theoretical guarantees of our privacy-preserving algorithm, in terms of convergence rate and privacy respectively. We present the results of our experimental study in Section 6 before surveying related work in Section 7 and concluding in Section 8.

2 Preliminaries

We start by describing the background and threat model considered in our work. Sections 2.1 and 2.2 introduce respectively general notations and the gossip-based averaging algorithm used by most DL algorithms. Section 2.3 describes some privacy attacks on DL and some existing countermeasures. Finally, Section 2.4 describes our threat model.

2.1 Decentralized learning

We consider a set of n nodes $\mathcal{V} = [\![1, n]\!]$, each owning a model of d parameters, whose aim is to solve a DL problem without sharing raw training data. While each node, $a \in \mathcal{V}$, stores a local data distribution $\mathcal{D}_a$, the goal is to determine the model parameters $x^* \in \mathbb{R}^d$ that optimize the learning problem over the local datasets of all participating nodes. This is done by minimizing an average loss function:

$$\underset{x\in\mathbb{R}^d}{\operatorname{argmin}} \left[f(x) = \frac{1}{n}\sum_{a=1}^{n} \underbrace{\mathbb{E}_{\xi\sim\mathcal{D}_a}\left[F_a(x;\xi)\right]}_{f_a(x)} \right], \tag{1}$$

where $f_a(x)$ represents the local objective function associated with node a, and $F_a(x;\xi)$ quantifies the prediction loss associated with the model parameters, x, for the sample, ξ, potentially encompassing non-convex characteristics.

To solve Equation (1), we proceed in T successive iterations, with each node, a, keeping its own local model, $x_a^{(t)}$, for each iteration, $t \in [\![0, T]\!]$. The goal is to make the averaged model, $\bar{x}^{(t)} := \frac{1}{n}\sum_{a=1}^{n} x_a^{(t)}$, converge to x^*.

The learning process involves collaborative interactions between nodes, which are connected by an underlying communication topology. At each iteration t, each node first trains its model on its local data and then aims to average it with the models of other nodes. During the averaging step, each node restricts its communication to its neighbors in the communication topology using gossip averaging (Section 2.2). Yet, sharing only model parameters may still leak sensitive information, thus hurting privacy.

2.2 Gossip averaging

Many DL algorithms rely on gossip averaging to estimate and share the average model $\bar{x}^{(t)} := \frac{1}{n}\sum_{a=1}^{n} x_a^{(t)}$ at each iteration t [10, 25]. A gossip-averaging operation can consist of multiple successive rounds. In each averaging round, s, the nodes communicate according to a *gossip matrix*, $W^{(t,s)}$, in the following manner. Each node,

a, sends a message, $m_{a\to v}^{(t,s)} \in \mathbb{R}^d$, containing the model parameters to each neighbor, v. Upon receiving it, node v weighs the received model using $W_{a,v}^{(t,s)}$ to perform averaging. In the simplest setting, $m_{a\to v}^{(t,s)}$ corresponds to the current local estimate of $\bar{x}^{(t)}$, this estimate is updated to $\sum_{v\in\mathcal{V}} W_{a,v}^{(t,s)} m_{v\to a}^{(t,s)}$, and it converges to $\bar{x}^{(t)}$ as s tends to infinity. We make the following assumption on $W^{(t,s)}$:

Assumption 2.1. *All gossip matrices are* symmetric, ${}^{\top}W^{(t,s)} = W^{(t,s)}$ *and* stochastic, $\forall a \in \mathcal{V}, \sum_{v\in\mathcal{V}} W_{a,v}^{(t,s)} = 1$.

While the symmetry assumption is not always necessary [10, 25], it is a common assumption for complexity proofs that enables tighter bounds [9, 23]. In our case, it enables convergence and privacy analysis.

We also denote by $\Gamma_a^{(t,s)}$ the set of neighbors to which node a sends its model, and by $d_a^{(t,s)}$ the corresponding degree of a. Formally, we have $\Gamma_a^{(t,s)} := \{v \in \mathcal{V} \mid W_{v,a}^{(t,s)} \neq 0\}$. Note that, due to Assumption 2.1 the networks are symmetric: $v \in \Gamma_a^{(t,s)} \iff a \in \Gamma_v^{(t,s)}$. Moreover, we consider $\Gamma_a^{(t,s)}$ to be a *closed neighborhood* unless otherwise specified, i.e. $a \in \Gamma_a^{(t,s)}$. A node may thus send virtual messages to itself. This property will be pivotal to our approach and is common for efficient averaging schemes [41].

Finally, several averaging approaches add a mask [5] or noise [9] to the messages to protect the privacy of the nodes' data. In this paper, we focus on noise-based approaches as they require less coordination between nodes and are more resilient to collusion between attackers.

Remark 2.2. *In DL, the averaging step does not need to reach exactly the same model at each node. Therefore, the rounds can be stopped before full convergence. In Zip-DL, even one round is sufficient ($s = 1$). Thus, in the rest of the paper, we will omit s in equations related to the averaging process.*

2.3 Privacy in Decentralized Learning

Privacy attacks. Numerous privacy attacks target Machine-Learning systems [7, 15, 37, 42, 47]. Most of these focus on attacking individual models or gradients, leaving attacks that exploit multiple models, such as those shared in DL, relatively underexplored. While some studies address this gap [31, 37], they often rely on strong assumptions about the attacker's capabilities or incur significant computational costs, especially in decentralized scenarios. To address these limitations, we adopt in this work two levels of MIA to analyze our approach: a *threshold*-based attack [7, 37], and a *classifier*-based MIA [45], which offers a computationally efficient and practical approach to evaluating privacy vulnerabilities in decentralized learning. Given some input sample x, MIAs aims to decide whether x belongs to a node's training set or not. Intuitively, if an attacker can accurately infer this information, it may be able to reconstruct part of a node's training dataset. In the following, we consider an attacker co-located with one of the participating nodes (the attacking node) that observes the models it receives in order to perform an MIA targeting the training set of some other distant node (the victim node). (See Section 2.4 just below for more detail.)

Differential Privacy. Introduced in databases, Differential Privacy (DP) provides a widely used framework for protecting models from such attacks [9, 12, 13, 30, 37]. In a decentralized scenario, DP can be instantiated in different ways. The most well-studied variants are *local differential privacy* (LDP) [22] and *central differential privacy*. The former assumes a local model without the existence of any trusted entity (*e.g.*, a server) to curate the noise that, in turn, provides LDP guarantees. However, this approach is usually detrimental for the utility. The latter only provides guarantees on a final, averaged model and relies on a trusted central server for adding the noise. It has been shown that the optimal tradeoff in both cases differs by a factor of n, the number of nodes [11]. To bridge this gap, relaxations of the strict scenario of LDP have been proposed [2, 9]. These relaxations include PNDP [9], which we consider in this work and detail in Section 5.

2.4 Threat model

We aim to protect the privacy of users data against *honest-but-curious* participating nodes during training. This scenario is in line with related work [4, 9, 15] in the domain of privacy-preserving DL, where the attacker can observe information about a victim node during training but does not deviate from the algorithm. We consider the attacker to be a node (or a set of nodes) participating in the training algorithm, but this can be extended to the case where an attacker is eavesdropping on a node's communication. The attacker's goal is to infer about the victim's data, which we quantify in terms of PNDP (see Section 5 for a formal definition). This notion of privacy in the context of DL is driven by the observation that privacy loss is not equal between all nodes in a distributed algorithm: close neighbors in the communication topology will receive more information from a node than nodes that are further away.

With PNDP in mind, our approach perturbs the models exchanged between nodes during training to prevent some honest-but-curious attacker from inferring precise information on a victim's node training distribution. In this setting, the attacker (which is co-located with one of the nodes) only has a partial view of the messages exchanged within the network, and its attacking power depends on its distance from the victim node within the communication graph. This is captured formally in Definition 5.2 in Section 5.1. In this setting, the attacker may never be in a position to reconstruct the final average model with precision, as its own local model might be biased by its position in the graph, non independent-and-identically-distributed (non-IID) data, and noise injected during the training process. The limited information that nodes can access in this setup and the key influence of their position on the (local) model they obtain is sufficient for most applications as the resulting local models remain valuable, even if they differ from node to node. Note, however, that if the goal of the DL algorithm is to produce a unified global model to use by some downstream application, one may apply central-DP or local-DP to global model before release to obtain DP guarantees also for this downstream application. Such a threat model is widespread in the literature that focuses on privacy-preserving DL such as [4, 9].

To empirically evaluate the performance of Zip-DL compared to its baselines, we conduct experiments with two paradigms of MIA that consider an attacker with different levels of knowledge of the victim's training set. The goal is to use a victim's message to infer whether a particular training sample was used to train the

Algorithm 1 Zip-DL-averaging for a node a at time t.

Input: local model x_a, stepsize γ, privacy parameter ς_a.
Output: Localized model average with correlated noise.

1: Get the gossip weights $W_a^{(t)}$, $d_a^{(t)} \leftarrow |\Gamma_a^{(t)}|$
2: Draw $Y_{a\to v}^{(t)} \sim \mathcal{N}(0, \gamma^2\varsigma_a^2)$ for $v \in \Gamma_a^{(t)}$
3: $Z_{a\to v}^{(t)} = Y_{a\to v}^{(t)} - \frac{1}{d_a^{(t)} W_{a,v}^{(t)}} \sum_{j\in\Gamma_a^{(t)}} W_{a,j}^{(t)} Y_{a\to j}^{(t)}$
4: **for all** $v \in \Gamma_a^{(t)}$ **do**
5: Send $x_a^{(t)} + Z_{a\to v}^{(t)}$ to v
6: Receive $x_v^{(t)} + Z_{v\to a}^{(t)}$ from v
7: **end for**
8: **return** $\sum_{v\in\Gamma_a^{(t)}} W_{a,v}^{(t)} (x_v^{(t)} + Z_{v\to a}^{(t)})$

victim's model in order to demonstrate how the formal privacy guarantees provided by Zip-DL complement with the empirically mounted MIA to capture the essence of real-world applications of our approach. More details are given in Section 6.1.

3 Zip-DL: Locally-Correlated Noise

3.1 Zip-DL in a nutshell

Gossip averaging typically requires multiple averaging rounds to provide a good estimate of the average of nodes' individual inputs [20]. Unfortunately, since averaging is required at each learning iteration, these rounds add up to a substantial network cost.

We drastically reduce this overhead by performing a single averaging round per learning iteration. Without noise, the cumulative effect of one-round averaging between each gradient-descent step is enough to ensure convergence [10, 25, 44].

Zip-DL adds noise to this process to provide PNDP guarantees. As one-round averaging is limited to a node's neighbors, the residual noise in partially averaged models remains high, which may disrupt learning and affect utility. We mitigate this effect by *correlating* the injected noise such that it sums to zero over each node's *closed neighborhood*. The correlation is local and eschews any coordination between neighbors.

In the following, we first detail the one-round localized averaging that lies at the core of Zip-DL (Algorithm 1), before moving on to the resulting decentralized SGD learning algorithm (Algorithm 2). We then state some fundamental properties of Zip-DL's global average model in Section 3.3.

3.2 Detailed description of Zip-DL

Zip-DL's model-averaging procedure is described in Algorithm 1. It relies on a stochastic communication topology [10] captured by the gossip matrix $W^{(t)}$, where t denotes the current learning iteration (Section 2.2). Node a first determines its neighborhood $\Gamma_a^{(t)}$ and the weights $W_a^{(t)}$ that its neighbors apply. Then, to protect its local data, a node a adds noise $Z_{a\to v}^{(t)}$ to its model $x_a^{(t)}$ before sending it to each of its neighbors, $v \in \Gamma_a^{(t)}$. By construction, the added noise sums to zero (Lines 2-3 of Algorithm 1) so as not to affect the computation of the global average. A node adapts how it protects its data by picking its own privacy parameter ς_a, which itself drives the variance $\gamma^2\varsigma_a^2$ of the injected noises.

Algorithm 2 Zip-DL for a node a.

Input $x_a^{(0)}$ the initial model, T the number of iterations.

1: **for** $t = 0$ to $T-1$ **do**
2: Draw $\xi_a^{(t)} \sim \mathcal{D}_a$, compute $g_a^{(t)} := \nabla F_a(x_a^{(t)}, \xi_a^{(t)})$
3: $x_a^{(t+1/2)} = x_a^{(t)} - \gamma g_a^{(t)}$
4: $x_a^{(t+1)} =$ Zip-DL-averaging$(x_a^{(t+1/2)}, \gamma, \varsigma_a)$
5: **end for**

To generate zero-summing noises in Algorithm 1, a node a first generates an *initial noise* $Y_{a\to v}^{(t)}$ (Line 2) for each of its neighbors v. Those noises are then correlated to create *pairwise noises* $Z_{a\to v}^{(t)}$ (Line 3) that will directly be added to the model sent to each neighbor. Those pairwise noises are the ones observed by an attacker.

In contrast to [14], Algorithm 1 uses a *closed neighborhood* that includes the local node a (*i.e.*, $a \in \Gamma_a$). Hence, even if a is surrounded by attackers after an eclipse attack [38], a's model remains protected to some extent as the noises of the models sent to $\Gamma_a \setminus \{a\}$ do not cancel out.

Zip-DL's main algorithm (Algorithm 2) is a DL algorithm. At each iteration t, each node a first performs a local gradient step on its local model $x_a^{(t)}$ to produce an intermediate model $x_a^{(t+1/2)}$ (Lines 2-3). The local model for the next iteration, $x_a^{(t+1)}$, is then obtained by applying Zip-DL's averaging procedure (Algorithm 1) to this model $x_a^{(t+1/2)}$.

3.3 Zip-DL's core properties

The following results pave the way for the formal analysis of Zip-DL in Section 4. If there is no influence of the time factor, we remove the (t) superindex to alleviate the notation (e.g. when a lemma holds for all $t \in [\![0, T]\!]$). Proofs that are not provided in this section can be found in Appendix C.

First, we state a property that summarizes the effect of the noise generated by a node on the network:

Lemma 3.1. Noise cancellation on the global model. *For every node $a \in \mathcal{V} = [\![1, n]\!]$, it holds that*

$$\sum_{v=1}^{n} W_{a,v} Z_{a\to v} = 0 = \sum_{v=1}^{n} W_{v,a} Z_{a\to v}.$$

This lemma states that a node does not add noise to the overall network, and leads to the following crucial corollary.

Corollary 3.2. Impact on the global average model. *For every epoch $t \in [\![0, T]\!]$, we have:*

$$\bar{x}^{(t+1)} = \bar{x}^{(t+1/2)}.$$

While simple, this corollary is pivotal in our convergence analysis of $\bar{x}^{(t)}$. Without this property, the bound on the expectation of $\left\|\bar{x}^{(t+1)} - x^*\right\|^2$ suffers from an extra term because of the noise.

Finally, Lemma 3.3 describes the behavior of the pairwise noise generated by Zip-DL: it follows a Gaussian distribution, which is standard for deriving formal privacy guarantees.

Lemma 3.3. Noise characterization for Algorithm 1. *Consider that for node a, for all $v \in \Gamma_a^{(t)}$, $Y_{a\to v}^{(t)} \sim \mathcal{N}\left(0, \gamma^2\varsigma_a^2\right)$, for a fixed topology*

$W^{(t)}$. *Then, using the definition of Algorithm 1, we have:*

$$\forall a, v \in [[1, n]], Z_{a\to v}^{(t)} \sim \mathcal{N}\left(0, (\sigma_{a\to v}^{(t)})^2\right)$$

with

$$(\sigma_{a\to v}^{(t)})^2 = \left(\frac{(d_a-1)^2}{d_a^2} + \frac{\sum_{j\in\Gamma_a^{(t)}, j\neq v}(W_{a,j}^{(t)})^2}{(d_a W_{a,v}^{(t)})^2}\right)\gamma^2\varsigma_a^2.$$

Note that Lemma 3.3 entails that the variance of the noise added to sent messages is strongly linked to the communication topology. This means that the chosen communication topology influences the privacy of our system, which further motivates our use of PNDP Section 5.1.

Remark 3.4. *When considering an k-regular topology or even a topology where only the incoming degree is fixed at k for all the nodes with a uniform weight distribution [10], then for a node a, we have $(\sigma_{a\to v}^{(t)})^2 = \frac{k-1}{k}\gamma^2\varsigma_a^2$. If we fix the same privacy parameter ς_a for all nodes, the noises generated by individual nodes all follow the same distribution.*

4 Convergence of Zip-DL

We now analyze the convergence rate of Zip-DL. The proof of the results stated in this section follows a similar structure to that of [23]. Detailed versions of proofs related to this section can be found in Appendix D.

Section 4.1 describes the assumptions we used for the convergence proof. Then, Section 4.2 details our bound in the setting described.

4.1 Assumptions

To ensure convergence, we define some assumptions that are common in the literature and that mostly follow those of [23]. First, we make assumptions about the smoothness and convexity of the loss functions:

Assumption 4.1. (L-smoothness). *The functions $F_i : \mathbb{R}^d \times \Omega \to \mathbb{R}$ are differentiable for each $i \in \mathcal{V}$ and $\xi \in supp(\mathcal{D}_i)$, and there exists a constant $L \geq 0$ such that for each $x, x' \in \mathbb{R}^d$ and $\xi \in supp(\mathcal{D}_i)$:*

$$\left\|\nabla F_i(x', \xi) - \nabla F_i(x, \xi)\right\| \leq L\left\|x - x'\right\|. \tag{2}$$

Assumption 4.2. (μ-convexity) *Each function f_i is μ-convex for a constant $\mu \geq 0$. For all $x, x' \in \mathbb{R}^d$:*

$$f_i(x) - f_i(x') + \frac{\mu}{2}\left\|x - x'\right\|_2^2 \leq \left\langle \nabla f_i(x),\, x - x'\right\rangle.$$

We also assume the noise caused by stochastic gradient descent (SGD) is bounded. This is particularly important since we consider a possible non-IID data distribution:

Assumption 4.3. (Bounded noise at the optimum) *Consider x^* such that $x^* := \operatorname{argmin} f(x)$ and define*

$$\vartheta_i^2 := \left\|\nabla f_i(x^*)\right\|^2, \qquad \bar{\vartheta}^2 := \frac{1}{n}\sum_{i=1}^n \vartheta_i^2. \tag{3}$$

In addition, define

$$\omega_i^2 := \mathbb{E}_{\xi_i}\left[\left\|\nabla F_i(x^*, \xi_i) - \nabla f_i(x^*)\right\|_2^2\right] \tag{4}$$

and $\bar{\omega}^2 := \frac{1}{n}\sum_{i=1}^n \omega_i^2$. Then $\bar{\vartheta}^2$ and $\bar{\omega}^2$ are bounded.

Intuitively, $\bar{\vartheta}^2$ measures the noise level and $\bar{\omega}^2$ the diversity of the locally sampled functions f_i. It is important to note that $\bar{\omega}^2$ is strongly linked to the data distribution. In particular, it will tend to be larger in a non-IID setting.

Finally, we state the assumption on the mixing matrix:

Assumption 4.4. (Expected consensus rate) *There exists $p \in [0, 1]$ such that for all matrices $X \in \mathbb{R}^{d\times n}$ and all iteration $t \in [[0, T]]$, if we define $\bar{X} := \frac{1}{n}X1_{n\times n}$ where $1_{n\times n} \in \mathbb{R}^{n\times n}$ is the matrix composed of ones, we have*

$$\mathbb{E}_{W^{(t)}}\left[\left\|W^{(t)}X - \bar{X}\right\|_F^2\right] \leq (1-p)\left\|X - \bar{X}\right\|_F^2.$$

This assumption is standard in the decentralized consensus literature, with p a value linked to the spectrum of $\mathbb{E}\left[{}^\intercal W^{(t)}W^{(t)}\right]$ [6].

4.2 Convergence rates of Zip-DL

We now state the formal convergence of Zip-DL in the strongly convex case:

Theorem 4.5. Convergence rate of Zip-DL. *For any number of iterations T, there exists a constant stepsize γ s.t. for Algorithm 2, $\frac{1}{2W_T}\sum_{t=0}^T w_t(\mathbb{E}\left[f(\bar{x}^{(t)})\right] - f^*) + \frac{\mu}{2}r_{T+1}$ is bounded by:*

$$O\left(\frac{\bar{\omega}^2}{n\mu T} + \frac{LA'}{\mu^2T^2} + \frac{r_0L}{p}\exp\left[-\frac{\mu p(T+1)}{192\sqrt{3}L}\right]\right),$$

where $A' = \frac{16-4p}{2(16-7p)}(\bar{\omega}^2 + \frac{18}{p}\bar{\vartheta}^2) + \frac{d}{n}\frac{16-4p}{16-7p}\sum_{a,v=1}^n d_a\frac{(d_v-1)^2}{d_v}\varsigma_v^2$, $f^* = f(x^*)$, $r_t = \mathbb{E}\left[\left\|\bar{x}^{(t)} - x^*\right\|^2\right]$, $w_t = (1-\frac{\mu}{2}\gamma)^{-(t+1)}$ and $W_T = \frac{1}{T}\sum_{t=1}^T w_t$.

Or, if we prefer a formulation to reach a desired accuracy:

Corollary 4.6. *Setting all the constants to be the same as in Theorem 4.5, for any target accuracy $\rho > 0$, there exists a constant stepsize γ such that Algorithm 2 reaches the target accuracy after at most*

$$\frac{3\kappa\bar{\omega}^2}{n\mu\rho} + \sqrt{\frac{3\kappa LA'}{\rho\mu^2}} + \frac{192\sqrt{3}L}{\mu p}\ln\left[\frac{3\kappa r_0 L}{\rho p}\right]$$

training iterations, where κ is the constant that arises when upper bound $O\left(\frac{\bar{\omega}^2}{n\mu T} + \frac{LA'}{\mu^2T^2} + \frac{r_0L}{p}\exp\left[-\frac{\mu p(T+1)}{192\sqrt{3}L}\right]\right)$ is expanded out.

This bound is similar to the one of [23]. The first and last terms are the same, except for the constants in the logarithm, which do not influence overall convergence since the logarithmic term is the slowest to grow. The second term however contains the additional complexity of our approach, in particular in the definition of A'. Our additional term is of the form $\sqrt{\frac{3\kappa Ld(16-4p)}{2n(16-7p)\mu^2\rho}\sum_{a,v=1}^n d_a\frac{(d_v-1)^2}{d_v}\varsigma_v^2}$. This term is weighted by $\rho^{-\frac{1}{2}}$ and is not the one that grows fastest as ρ goes to 0, proving the limited impact of our approach on convergence. We observe that this term contains a weighted average of the noise propagated by every node, showing the intuitive behavior of slowing down convergence if the noise ς_a^2 becomes too big. Interestingly, this term grows as the network size or density grows. Indeed, the higher the degree, the more the noise injected at each iteration, and the larger the network, the longer it takes for the noise to propagate and cancel out.

We can also compare this bound to a recent noisy approach [2], even if their privacy setting is different from ours. While they do not consider a strongly-convex scenario like us and assume a weaker assumption that is implied by a strongly-convex property, we observe that the noise variance appears on their leading term, in $O(\frac{1}{T})$. The analysis we performed here on an algorithm without noise cancellation would also have yielded similar results. On the other hand, our approach delegates the impact of the noise to the second leading term, yielding faster convergence rates.

Similarly, the bound presented in [9] is also affected in its leading term by the factor σ^2/T, where σ denotes the DP noise constant, as established in Theorem 10 of [9]. Consequently, the same conclusion drawn in the previous paragraph can be applied in this context.

Relaxation of assumptions. Following [23], we conjecture that our proof can be generalized to the convex and the non-convex scenarios, thus weakening Assumption 4.2. In particular, the difficulty of adapting to a non-convex scenario mostly lies in the gradient descent analysis, which is only marginally modified by our approach. We chose to keep to the strongly convex scenario because our direct baseline also made such an assumption [9].

Likewise, we conjecture it is possible to loosen Assumption 4.4 by adopting the same approach as in [23]. However, we chose to stick to a more standard assumption, as it was not the main focus of this work.

Node dropout. The formal analysis of convergence of Zip-DL relies on the noises canceling out on average (Lemma 3.1). In practice, nodes in DL may have intermittent availability, *i.e.*, they may join or leave the network at any time. As a result, the injected noise in Zip-DL may not always sum to zero. However, the inherent stochasticity of the training process and the robustness of gradient-based optimization mitigate the impact of node dropouts in Zip-DL. We experimentally demonstrate the resilience of Zip-DL to node dropouts in Section A.5 and discuss the possible adaptation of our convergence proof to such scenarios.

Sketch of proof. (*Theorem 4.5*). We mostly follow the proof of [23]. The main challenge lies in adapting the set of lemmas to our noisy approach. The mini-batch variance (Proposition D.3) is unchanged, as it only relies on hypotheses on the loss function, which are identical to ours. The descent lemma (Lemma D.4) is where Corollary 3.2 comes into play, since canceling noises have no impact on the averaged model. Without noise cancellation, an additional term would have been added here, which would have propagated to the leading term of the convergence rate in $\frac{1}{T}$.

Finally, the recursion for consensus distance (Lemma 4.7) is modified because of the noise addition, which becomes an extra term. In addition to this extra term, our main recursion is slightly altered, with an additional factor to the recursive term. While this additional factor prevents solving the main recursion directly, a manipulation leads to a term that can be solved, yielding the desired result. □

This proof relies on three main lemmas detailed in Appendix D. Two of them remained unchanged using Zip-DL's properties. For the sake of completeness, we state the adapted lemma that presents an additional last term compared to state-of-the-art DL [23]. This term arises from noises shifting local models from the true average.

Lemma 4.7. (Recursion for consensus distance) *Under Assumptions 4.1 to 4.4, if stepsizes $\gamma \leq \frac{p}{96\sqrt{3}L}$, then for any $\beta > 0$:*

$$\begin{aligned}\Xi_t \leq\ & (1+\beta)\left(1-\frac{7p}{16}\right)\Xi_{t-1} + \gamma^2(1+\beta)\left(\bar{\omega}^2 + \frac{18}{p}\bar{\vartheta}^2\right)\\ & + (1+\beta)\,\frac{36L}{p}\left(f(\bar{x}^{(t-1)}) - f(x^*)\right)\\ & + \gamma^2(1+\beta^{-1})\frac{d}{n}\sum_{i=1}^{n} d_i \sum_{v=1}^{n}\left(\frac{(d_v-1)^2}{d_v}\varsigma_v^2\right),\end{aligned}$$

where $\Xi_t = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_t\left[\left\|x_i^{(t)} - \bar{x}^{(t)}\right\|^2\right]$ is the consensus distance

5 Pairwise Network Differential Privacy

We now formalize the privacy guarantees of Zip-DL in terms of *pairwise-network differential privacy (PNDP)*, a graph-based variant of DP introduced in the work by Cyffers et al. [9] to capture the unique threats to privacy introduced by the DL framework. This section establishes the formal PNDP guarantees that Zip-DL provides and dissects further its analytical properties.

More concretely, we first present the additional assumption and privacy definition used in the analysis (Section 5.1), before defining an equivalent formulation of our algorithm (Section 5.2). Section 5.3 will then exploit this formulation to express the evolution of the system, which is pivotal to our privacy analysis of Zip-DL presented in Section 5.4. We finally consider the simpler case (Algorithm 1 in Section 5.5), and link our results to those of [9].

5.1 Assumptions & definitions

When discussing PNDP, we use the same notations and definitions as [9]. Specifically, with $\mathcal{D} = (\mathcal{D}_a)_{a\in\mathcal{V}}$ denoting a set of datasets across all the nodes, we call a pair of (entire) datasets $\mathcal{D}$ and $\mathcal{D}'$ *adjacent*, denoted by $\mathcal{D} \sim_a \mathcal{D}'$, if there is some node and only one node $a \in \mathcal{V}$ for which $\mathcal{D}_a$ and $\mathcal{D}'_a$ differ. Considering two *adjacent* datasets is the first building block to express differential privacy properties.

We analyze how Zip-DL guarantees PNDP for an input dataset $\mathcal{D}$ (a given initial data distribution between the nodes). To this purpose, we require two additional assumptions, in addition to those highlighted in Section 4.1. First, we need the distance between the models trained on two adjacent datasets to be bounded, which aligns with Assumption 1 in [9].

Assumption 5.1. *There exists some constant $\Delta > 0$ such that for any adjacent datasets $\mathcal{D} \sim_a \mathcal{D}'$, we have*

$$\sup_{x\in\mathbb{R}^d}\sup_{\xi,\dot{\xi}\in\mathcal{D}\times\mathcal{D}'}\left\|\nabla F(x,\xi) - \nabla F(x,\dot{\xi})\right\|^2 \leq \Delta^2. \tag{5}$$

This is a standard assumption when considering differentially private algorithms: we use a bound on the original perturbation and observe how this perturbation can be scaled by the algorithm. This assumption is typically enforced through *gradient clipping* [1]. Due to space constraints, an analysis of its impact on Zip-DL is deferred to Section A.3.

For a pair of adjacent datasets, Muffliato [9] introduces the notion of *privacy view* on two such datasets:

Definition 5.2. [9] The *privacy view* of a node v after T steps for a dataset $\mathcal{D}$ is:

$$O_v(\mathcal{A}^{(T)}(\mathcal{D})) = \{m_{w\to v}^{(t)} \mid t \in [\![1,T]\!], v \in \Gamma_w^{(t)}\} \cup \{x_v\},$$

with $\mathcal{A}^{(T)}$ a state-sharing algorithm iterated T times such as Algorithm 1 or Algorithm 2, and $\mathcal{A}^{(T)}(\mathcal{D})$ the set of all messages sent by neighboring nodes on the network during the execution of the algorithm.

The privacy view represents a projection from the set of all the messages in an execution $\mathcal{A}^{(T)}$ to the set of messages that v receives during the algorithm's execution.

When considering this privacy view $O_v(\mathcal{A}^{(T)}(\mathcal{D}))$, we consider the scenario where node v would be an *honest-but-curious* attacker and tries to infer information from its observations — the privacy view. This view is then used to define PNDP [9], by leveraging the definition of Rényi-DP [30].

Definition 5.3. (Pairwise Network Differential Privacy) For $g : \mathcal{V}^2 \to \mathbb{R}^+$ and $\alpha > 1$, a mechanism $\mathcal{A}^{(T)}$ satisfies (α, g)-*Pairwise Network Differential Privacy (PNDP)* if, for all pairs of distinct nodes $a, v \in \mathcal{V}$ and adjacent datasets $\mathcal{D} \sim_a \mathcal{D}'$, we have

$$D_\alpha\left(O_v(\mathcal{A}^{(T)}(\mathcal{D}))\|O_v(\mathcal{A}^{(T)}(\mathcal{D}'))\right) \leq g^{(T)}(a,v),$$

where $D_\alpha\,(P\|Q)$ is the Rényi divergence [16] between probability distributions P and Q:

$$D_\alpha\,(X\|Y) = \frac{1}{\alpha-1}\ln\int\left(\frac{\mu_X(z)}{\mu_Y(z)}\right)^\alpha \mu_Y(z)dz,$$

with μ_X and μ_Y the densities of X and Y.

Therefore, $g^{(T)}(a,v)$ quantifies the *privacy leaked* from a to v, and our goal is to constrain it to a minimal value. This decentralized approach harnesses communication topology, in contrast to DP or Renyi-DP, thus fully exploiting the specificity of a decentralized context.

The choice of this privacy guarantee is further motivated by the synergy between Rényi-DP and Gaussian noise [30], as the following lemma underlines:

Lemma 5.4. *[16] Suppose that $X \sim \mathcal{N}(\mu_X, \Sigma)$ and $Y \sim \mathcal{N}(\mu_Y, \Sigma)$. Then for all $\alpha > 1$, we have:*

$$D_\alpha\,(X\|Y) = \frac{\alpha}{2}{}^\intercal(\mu_X-\mu_Y)\Sigma^{-1}(\mu_X-\mu_Y). \tag{6}$$

This lemma is the key motivation to our use of Gaussian noises in our approach: we require an additivity property to generate cancelling noises, as well as differential privacy properties. Thus, Gaussian noise is a natural candidate, as it fits both criterions.

Rényi-divergence usually provides important properties when considering privacy concerns. Most notably, the *composition theorem* and the preservation by *post-processing* [30]. Of those two, the former allows for an easy way to derive the privacy guarantee of the composition of differentially private algorithms. When considering a process with multiple rounds, this makes it practical to compose privacy guarantees between rounds and significantly alleviates the analysis.

Remark 5.5. *Since we consider a projection of the set of all messages $\mathcal{A}^{(T)}(\mathcal{D})$ on the view of the attacker, we cannot naively apply composition theorems on $O_v(\mathcal{A}^{(t)}(\mathcal{D}))$ to this approach directly. That is because here, the composition would rely on external information, that was not in the view of the attacker. To circumvent this, the original paper [9] considers a full averaging algorithm, meaning composition can be performed by using the (common) final state of the averaging algorithm.*

However, we want a more usual view of DL, where we alternate between one round of averaging and one round of gradient descent. To avoid using composition, we must be able to analyze the behavior of the noise through the gradient. To this end, we consider the following assumption:

Assumption 5.6. *For all $i \in \mathcal{V}$, for all data sample ξ_i and model x, if we consider a noise $Z \sim \mathcal{N}(0,\Sigma)$, then we have:*

$$\nabla F_i(x+Z,\xi_i) \sim \mathcal{N}(\nabla F_i(x,\xi_i), L\Sigma).$$

In essence, Assumption 5.6 implies that the gradient of a model perturbed with Gaussian noise stays close to the unnoised (original) gradient while following a Gaussian distribution around this unnoised gradient. The range of the standard deviation is bounded by the smoothness constant L (Assumption 4.1), which comes from the remark that $\left\|\nabla F_i(x+Z,\xi_i) - \nabla F_i(x,\xi_i)\right\|^2 \leq L\,\|Z\|^2$. This assumption will allow us to simplify privacy expressions without resorting to a composition theorem. Most notably, Lemma 5.8 links an execution of Zip-DL to an execution of decentralized learning without any noise. This link will be pivotal to the privacy proof. We further evaluate Assumption 5.6 in Section A.6.

5.2 Equivalent system formulation

Gossip matrices (Section 2.2) are a natural tool to analyze how information propagates in a communication graph over several communication rounds. Unfortunately, they cannot be directly applied to Algorithm 2, as they assume that each node sends the same information to all its neighbors in a given round. This assumption does not hold for Algorithm 2, where the noise $Z_{a\to v}$ added by each node a to its model during the Zip-DL-averaging step (line 5 of Algorithm 1) is different for each of a's neighbors.

We overcome this difficulty by considering an equivalent virtual communication graph of n^2 nodes that emulate the behavior of the n nodes executing Algorithm 2. In this construction, each original node $a \in \mathcal{V}$ is replaced by n virtual nodes $a_1, .., a_n \in \hat{\mathcal{V}}$ connected in a clique. Each virtual node a_v is then connected to v_a in the virtual communication graph if a is connected to v.

This emulated network makes it possible to track the privacy loss incurred by our algorithm, whose behavior can be interpreted as a sequence of linear matrix operations on the states of the virtual nodes. Because each virtual node replicates the state of its real node, the system's state is encoded in a matrix of dimension $n^2 \times d$, while message exchanges and state updates are captured by matrices of size $n^2 \times n^2$ (since the virtual communication topology contains n^2 nodes).

In the remainder of this section, we present in more detail the entities we use to analyze the privacy loss of Algorithm 2 using virtualization. Virtual entities are decorated with the symbol $\hat{}$: if A describes an object in the original system, then $\hat{A}$ represents its

counterpart in the virtual topology. We note $\hat{\mathcal{V}} = [\![1, n^2]\!]$ the set of virtual nodes, where the real node i is represented by the virtual nodes ranging from $n(i-1)+1$ to ni. $\hat{X}^{(t)}$ represents the stacking of virtual models at time t, *i.e.*,

$$\hat{X}^{(t)} = {}^{\intercal}\left({}^{\intercal}x_1^{(t)}, \quad \ldots, \quad {}^{\intercal}x_1^{(t)}, \quad {}^{\intercal}x_2^{(t)}, \quad \ldots, \quad {}^{\intercal}x_n^{(t)}\right),$$

in which the local model $x_a^{(t)} \in \mathbb{R}^d$ is duplicated n times across all the virtual nodes associated with node a. $\hat{X}^{(t)} \in \mathbb{R}^{n^2 \times d}$ in the general case, and so do the noises generated by all the nodes. For simplicity when defining those noises, we focus in the following on the case $d = 1$ to introduce the notations, but the approach generalizes seamlessly to higher dimensions.

The noises generated in Algorithm 1 are captured by two random vectors $\hat{Y}^{(t)}$ and $\hat{Z}^{(t)}$ of dimension n^2, defined component-wise by

$$\hat{Y}^{(t)}_{n(i-1)+j} := Y^{(t)}_{i\to j}, \qquad \forall i, j \in \mathcal{V},$$
$$\hat{Z}^{(t)}_{n(i-1)+j} := Z^{(t)}_{i\to j}, \qquad \forall i, j \in \mathcal{V}.$$

Due to the definition in Algorithm 1, $\hat{Z}^{(t)}$ results from a linear combination of $\hat{Y}^{(t)}$:

$$\hat{Z}^{(t)} = \hat{C}^{(t)}\hat{Y}^{(t)}, \tag{7}$$

where, $\hat{C}^{(t)}$ is the block-diagonal matrix filled with 0 values, except in the following positions when a, v, j range over $\mathcal{V}$:

$$\hat{C}^{(t)}_{n(a-1)+v,n(a-1)+j} := \begin{cases} \frac{d_a - 1}{d_a} & \text{if } j = v \wedge v \in \Gamma_a^{(t)}, \\ -\frac{W^{(t)}_{a,j}}{d_a W^{(t)}_{a,v}} & \text{if } j \neq v \wedge v \in \Gamma_a^{(t)}, \\ 0 & \text{Otherwise.} \end{cases}$$

The covariance matrix of $\hat{Y}$ is the diagonal matrix in which each node's variance (ς_a^2) is repeated n times.

The covariance matrix of $\hat{Z}$ is $\Sigma^{(t)}_{\hat{Z}} = \hat{C}^{(t)}\Sigma_{\hat{Y}}{}^{\intercal}\hat{C}^{(t)}$ due to Equation (7).

From a given gossip matrix $W^{(t)}$, we construct $\hat{W}^{(t)}$ as the communication matrix where each virtual node only communicates with one fixed node. We also introduce $\hat{M}$, which mixes information between the virtual nodes afterward.

$$\hat{W}^{(t)}_{\hat{i},\hat{j}} := \begin{cases} W^{(t)}_{i,j}, & \text{if } \hat{i} = n(i-1)+j, \hat{j} = n(j-1)+i, \\ 0, & \text{Otherwise.} \end{cases}$$

$$\hat{M} := \begin{pmatrix} \mathbf{1}_n & \mathbf{0}_n & \mathbf{0}_n & \ldots & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{1}_n & \mathbf{0}_n & \ldots & \mathbf{0}_n \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \mathbf{0}_n & \mathbf{0}_n & \mathbf{0}_n & \ldots & \mathbf{1}_n \end{pmatrix} \in \mathbb{R}^{n^2 \times n^2}, \tag{8}$$

where $\mathbf{1}_n = [1]_{i,j\in[\![1..n]\!]}$ and $\mathbf{0}_n = [0]_{i,j\in[\![1..n]\!]}$ represent the matrices of dimension $n \times n$ full of ones or zeros, respectively. $\hat{M}$ creates a fully connected communication network between the virtual nodes of a given real node. In doing so it captures how each local node averages the individual models it receives through $\hat{W}^{(t)}$.

Using this matrix, we obtain the following virtual gossip round:

$$\hat{X}^{(t+1)} = \hat{M}\hat{W}^{(t)}(\hat{X}^{(t+1/2)} + \hat{Z}^{(t)}). \tag{9}$$

The following lemma ensures that the update rule stays the same as Algorithm 1 of Algorithm 1, proving we have constructed something equivalent to the non-virtual update rule:

Lemma 5.7. *Consider $i \in \mathcal{V}$ and $t \in \mathbb{N}$. Then we have:*

$$\forall k \in \mathcal{V}, \hat{X}^{(t)}_{ni+k} = X_i^{(t)}.$$

5.3 Accounting for noises over time

In order to track how privacy losses propagate from one SGD round to the next without using a composition theorem (see Remark 5.5), we further consider T successive rounds of Algorithm 2. These T rounds incur the generation of Tn^2 individual noise values at Line 2 of Algorithm 1. We track the correlation between these noises and the model parameters to which they are applied in the virtual system through covariance matrices of size tn^2, for $t \in [\![1, T]\!]$.

To track those $n^2 \times t$ noises, we consider matrices that aggregate data through time for notation purposes. Those matrices will be denoted by a $\tilde{}$ notation. Similarly to before, we consider $\tilde{Y}^{(t)} \in \mathbb{R}^{tn^2}$ a matrix stacking all the noises generated on the network.

Even if the noises at time $t+1$ are independent from the noises at time t, meaning the covariance matrix will be block-diagonal, we reach a simpler expression with time matrices. Formally, we have:

$$\Sigma_{\tilde{Y}^{(t)}} := \begin{pmatrix} \Sigma_{\hat{Y}} & \mathbf{0}_{n^2} & \ldots & \mathbf{0}_{n^2} \\ \mathbf{0}_{n^2} & \Sigma_{\hat{Y}} & \ldots & \mathbf{0}_{n^2} \\ \ldots & \ldots & \ldots & \ldots \\ \mathbf{0}_{n^2} & \mathbf{0}_{n^2} & \ldots & \Sigma_{\hat{Y}} \end{pmatrix} \in \mathbb{R}^{tn^2 \times tn^2}, \tag{10}$$

where $\Sigma_{\hat{Y}} \in \mathbb{R}^{n\times n}$ corresponds to the covariance matrix of the uncorrelated noises. This is a diagonal matrix. In the special case where all nodes have the same privacy parameter ς, then we have $\Sigma_{\tilde{Y}^{(t)}} = \varsigma^2 I_{tn^2 \times tn^2}$.

Using this and the decomposition $\hat{Z}^{(t)} = \hat{C}^{(t)}\hat{Y}^{(t)}$ (Equation (7)), where $\hat{Y}^{(t)} \sim \mathcal{N}(0, \Sigma_{\hat{Y}})$, we also create a decomposition $\tilde{Z}^{(t)} = \tilde{C}^{(t)}\tilde{Y}^{(t)}$, where $\tilde{C}^{(t)}$ a block diagonal matrix of all the $\hat{C}^{(t)}$.

For ease of notation, when considering matrices that aggregate through time, we will consider a constant communication matrix $W^{(t)} = W$. Our notations could be generalized at the expense of matrix product notations. For the temporal gossip matrix, we define the following:

$$\tilde{W}^{(T)} := \begin{pmatrix} (1-\gamma L)^T \\ \ldots, \\ (1-\gamma L) \end{pmatrix}\left((\hat{M}\hat{W})^T, \quad \ldots, \quad \hat{M}\hat{W}\right). \tag{11}$$

In particular, we have $\tilde{W}^{(T)} \in \mathbb{R}^{n^2 \times Tn^2}$. This matrix will appear in Theorem 5.9 and can be used to compute the propagation of the noise through the system after T steps.

This notation finally allows us to leverage Assumption 5.6. Using the equivalent formulation defined in Section 5.2, we now progress toward the privacy analysis. First, we derive the distribution of the model vectors:

Lemma 5.8. *Using Assumption 5.6, consider $\hat{\chi}^{(T)}$ a virtual execution without any noise, and every other source of randomness is the same. Then, we have:*

$$\hat{X}^{(T)} \sim \mathcal{N}\left(\hat{\chi}^{(T)}, L\hat{W}^{(T)}\tilde{C}^{(t)}\Sigma_{\tilde{Y}^{(t)}}{}^{\intercal}(\hat{W}^{(T)}\tilde{C}^{(t)})\right).$$

This lemma draws a parallel between an execution of Algorithm 2 and an unnoised execution and is at the core of our privacy analysis. Lemma 5.8 offers a structure to bound the Rényi divergence between

$\hat{X}^{(T)}$ on two executions on adjacent datasets. Its proof is deferred to Appendix B.

5.4 Zip-DL privacy analysis

We now focus on analyzing the formal privacy guarantees of Algorithm 2.

Theorem 5.9 (Privacy of Zip-DL). *T iterations of Zip-DL (Algorithm 2) satisfies $\left(\alpha, \epsilon^{(T)}(a,v)\right)$-PNDP, where $\epsilon^{(T)}(a,v)$ is bounded for any two nodes $a, v \in \mathcal{V}$ by:*

$$\frac{2\alpha\gamma^2\Delta^2}{L+4\gamma^2L^2}\sum_{t=0}^{T-1}\sum_{\substack{\hat{v}\in\hat{V}\\ \hat{w}\in\hat{\Gamma}_{\hat{v}}^{(t)}}}\frac{(2+4\gamma^2L)^t-1}{\left(\left(\tilde{W}\tilde{C}\right)^{(t)}\tilde{\Sigma}_{\tilde{Y}^{(t)}}{}^{\intercal}\left(\tilde{W}\tilde{C}\right)^{(t)}\right)_{\tilde{w},\tilde{w}}},$$

where $\tilde{\Sigma}_{\tilde{Y}^{(t)}}$ is a diagonal matrix representing the noise variances of all noises generated by the algorithm up to time T, $\tilde{C}^{(t)}$ is a block-diagonal matrix representing the correlation factor at each iteration t, and $\tilde{W}^{(t)}$ is the accumulation of all the powers of the gossip matrix defined in Section 5.2.

In essence, a node's privacy loss increases over time, and the influence of the privacy mechanism is denoted by the denominator: this term accounts for all the noises received by the virtual node $\hat{w}$. On the other hand, the numerator accounts for how models drift away from each other.

If we consider that all nodes have the same privacy parameter ς, then the denominator becomes akin to the norm of $(\tilde{W}\tilde{M})^{(t)}_{\hat{w}}$, which is similar to [9].

This result is a double sum over time and the attacker's neighbors, since in our notation $\hat{\Gamma}_{\hat{v}}^{(t)}$ is a set containing at most one value that translates whether w is in $\Gamma_v^{(t)}$ or not.

Remark 5.10. *This result naturally extends to colluding nodes if we consider $\hat{V} = \bigcup_{v\in V}\{n(v-1)+k \mid k \in \mathcal{V}\}$ to be the set of colluding nodes. We can thus have a similar bound of $\epsilon^{(T)}(a,V)$, for $V \subset \mathcal{V}$ a set of colluding nodes.*

Even if the matrices considered here are of large dimensions, this bound can be computed in practice since their underlying matrices are sparse: either they are diagonal by block, or some have only one element by line. For instance, both $\tilde{M}$ and $\tilde{\Sigma}_{\tilde{Y}^{(t)}}$ are diagonal by block since the noises generated at each iteration are independent.

Remark 5.11. *In practice, Assumption 5.6 may not always hold accurately. To capture the ripple effect of this inaccuracy and bound the privacy loss in this scenario, one may define an error term stemming from Assumption 5.6, using Corollary 4 of [30], and add the following term to Theorem 5.9:*

$$D_\infty\left(\hat{X}^{(T)} \| \mathcal{N}\left(\hat{\chi}^{(T)}, L\hat{W}^{(T)}\tilde{C}^{(t)}\Sigma_{\tilde{Y}^{(t)}}{}^{\intercal}(\hat{W}^{(T)}\tilde{C}^{(t)})\right)\right)$$

5.5 Zip-DL-avg privacy analysis

We also focus on the privacy of Algorithm 1 as a pure averaging algorithm. This removes gradient from the proof of Theorem 5.9, and thus Assumption 5.6 is not needed. By following the same proof with a simpler update rule, we can derive a more tractable term,

Theorem 5.12. *T iterations of Algorithm 1 satisfy $(\alpha, \epsilon^{(T)}(a,v))$-PNDP, where $\epsilon^{(T)}(a,v)$ is bounded for any two nodes $a, v \in \mathcal{V}$ by:*

$$\frac{\alpha\Delta^2}{2}\sum_{t=0}^{T-1}\sum_{\hat{v}\in\hat{V}}\sum_{\hat{w}\in\hat{\Gamma}_{\hat{v}}^{(t)}}\frac{\left((\hat{M}\hat{W})^T\right)_{\hat{w},\hat{a}}}{\left((\tilde{W}\tilde{C})^{(t)}\Sigma_{\tilde{Y}}{}^{\intercal}(\tilde{W}\tilde{C})^{(t)}\right)_{\tilde{w},\tilde{w}}},$$

where

$$\tilde{W}^{(T)} := \left((\hat{M}\hat{W})^T, \quad \ldots, \quad \hat{M}\hat{W}\right).$$

Remark 5.13. *This theorem generalizes the result of [9] by introducing the correlation matrix between all the generated noises $\tilde{C}^{(t)}$. Applied to the algorithm presented in [9], the correlation matrix $\tilde{C}^{(t)}$ in the above expression would instead be the identity matrix. Additionally, the numerator is also the same as the one of the original work, as we have $\left((\hat{M}\hat{W})^T\right)_{\hat{w},\hat{a}} = (W)^T_{w,a}$ where w, a are the nodes associated to the virtual nodes $\hat{w}, \hat{a}$.*

6 Evaluation

We evaluate Zip-DL on two practical learning tasks, image classification (on CIFAR-10) and movie recommendation (on MovieLens). We compare Zip-DL's performance[1] to that of two baselines: decentralized parallel stochastic gradient descent (D-PSGD) [26], and Muffliato [9], a privacy-preserving DL algorithm. The comparison focuses on two aspects: (i) the tradeoff between privacy (measured as the ROC-AUC of two membership inference attacks) and model utility (top-1 test accuracy on CIFAR-10, and test loss on MovieLens), and (ii) the cost of such privacy in terms of communication overhead. For the sake of completeness, we also consider True Positive Rate (TPR) at low False Positive Rate (FPR) rates for one of these attacks, in line with existing literature [7]. Those results are reported in Section 6.4.

6.1 Experimental setup

Communication graph. Throughout the evaluation, we use 100 nodes connected in a 6-regular communication graph. We assume all nodes are online and available at all times unless stated otherwise. The experiments with node dropouts are presented in Section A.5.

Baselines. We compare Zip-DL with two baselines: D-PSGD [26], a decentralized stochastic gradient descent algorithm without privacy guarantees (labeled *No noise* in the figures), and Muffliato, a state-of-the art privacy-preserving DL algorithm. To cover different communication settings, we consider two scenarios: *(i)* using 1 averaging round per training iteration, as in typical D-PSGD, and *(ii)* using 10 averaging rounds per training iteration, as recommended for Muffliato when applied to our network topology (Theorem 5 in [9]). In practice, using 10 averaging rounds ensures each node obtains an almost global average model.

First Learning Task — CIFAR-10. We evaluate Zip-DL and the two baselines on the image classification task of CIFAR-10 [24] using a Group Normalization (GN)-ResNet18 [18]. We opted for GN layers [40] over the traditional Batch Normalization layers due to their superior compatibility with differential privacy mechanisms, especially in decentralized scenarios [32, 43]. The training set comprises

[1] All code used in this section can be found at https://github.com/dimiarbre/ZIP-DL

50 000 data samples and the test set 10 000 data samples, spread uniformly between the nodes. The neural network has 11 189 312 trainable parameters. For utility, we consider the top-1 test accuracy for CIFAR-10.

Second Learning Task — MovieLens. We consider a recommendation task on the MovieLens dataset [17]. We use *movielens-small*[2], a dataset of 100 836 ratings from 610 users on 9742 movies, where each user has rated at least 20 movies. Given MovieLens is naturally partitioned between users, we allocate users to a given node. We use a matrix factorization model with the SGD optimizer. While it is similar to a classification task, it provides three new aspects in our experimental evaluation: *(i)* the underlying model is linear, *(ii)* the data is naturally non-IID, *(iii)* the model outputs numerical rating estimates (between 1 and 5). Because model outputs are numerical, we use the RMSE on predicted ratings (which corresponds to the task's test loss) as the utility metric for MovieLens.

Noise levels. Both Zip-DL and Muffliato use noise levels of the form $k\sigma$, with $k \in \{2^0, 2^1, \ldots, 2^8\}$ and σ such that $128\sigma = 0.225$. This range of values covers a broad set of behaviors for both Zip-DL and Muffliato on both tasks (CIFAR-10 and Movielens). Some additional, intermediary values of k are also used for relevant plots. More precisely, we directly use $k\sigma$ as the standard deviation of noise in Muffliato, while we derive a uniform privacy parameter ς_a from $k\sigma$ for Zip-DL using the formula in Remark 3.4. Doing so ensures that the standard deviation of pairwise noises $\sigma^{(t)}_{a\to v}$ is $k\sigma$ in both approaches.

Privacy attacks. We evaluate the privacy of the algorithms against an *honest-but-curious* attacker described in Section 2.4 using two membership inference attacks (MIA). We apply *(i)* a threshold-based attack [7, 37], and *(ii)* a more advanced classifier-based attack described in [45] and inspired by [33]. Both attacks seek to determine whether a victim node used a specific data point to train its local machine learning (ML) model. Taking an example as input, both attacks base their decisions on the example's loss computed by the node's local model, under the assumption that lower losses are indicative of training examples. The effectiveness of these attacks is evaluated using the Area Under the Curve (AUC) of the TPR plotted as a function of the FPR. (This curve is known as *Receiver operating characteristic* or ROC, hence the shorthand ROC-AUC.)

The threshold attack reaches its decision by comparing the example's loss obtained from the victim's final model to some fixed threshold. While simple, this approach establishes a baseline for privacy vulnerability: if such an attack proves successful, it implies that more sophisticated methods are likely to succeed as well [7].

While the threshold attack uses a single model, the classifier attack records multiple models shared by a single victim at different time stamps during the training process. For a given data example, the attacker considers the time series of the loss of this example on those models. The aim of the classifier is then to classify this time series as either a member or non-member of the victim node's local dataset. More precisely, the attack uses a Fully Connected Network (FCN) binary classifier with 2 hidden layers and uses the losses of 26 models obtained at fixed intervals during the victim's training process (including its first and final model). The binary classifier is trained on 70% of the victim's node local dataset (positive training examples) and 70% of the test data (negative training examples). Examples are re-weighted so that both classes have the same weight in the training. The attack is then evaluated on the remaining examples (local and test) not used for training. Because it requires a training phase with sufficient data, this attack assumes the attacker possesses considerable knowledge about the victim's dataset and is therefore much more aggressive than the threshold attack.

[2]https://files.grouplens.org/datasets/movielens/ml-latest-small-README.html

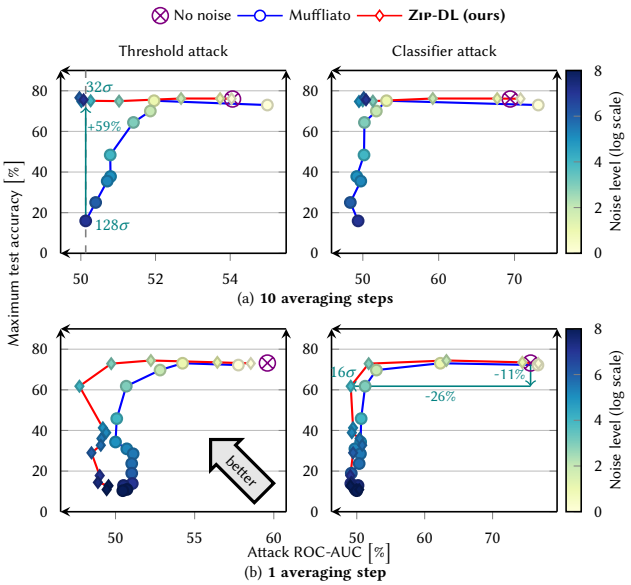

Figure 1: Maximum accuracy reached as a function of both attacks results on CIFAR-10. Color intensity represents a higher noise level. In all cases, the tradeoff is better for Zip-DL.

Utility-privacy trade-off and communication cost. In addition to each algorithm's privacy, we measure the best utility across models during training, computed on each task's test dataset (using top-1 accuracy for CIFAR-10, and RMSE on MovieLens MovieLens). We further measure the overall communication cost of the entire training process (measured in Teribytes). The utility and privacy measures are averaged over all nodes.

6.2 Zip-DL privacy-utility tradeoff

Figures 1 and 2 show the privacy-utility trade-off of Muffliato (blue curve) and Zip-DL (red curve) for multiple noise levels (represented by the filling color of the data points). The privacy-utility trade-off of D-PSGD ("No noise", i.e. no privacy protection) is shown for reference as a purple cross in a circle. Figure 1 charts the results obtained on CIFAR-10, while Figure 2 plots those obtained with MovieLens.

CIFAR-10 Results. In Figure 1, the average test accuracy across nodes is shown vertically (higher is better) and the attack's success (ROC-AUC) horizontally, with lower (and better) values on the left. A 50% ROC-AUC indicates the attack has been neutralized, as in this

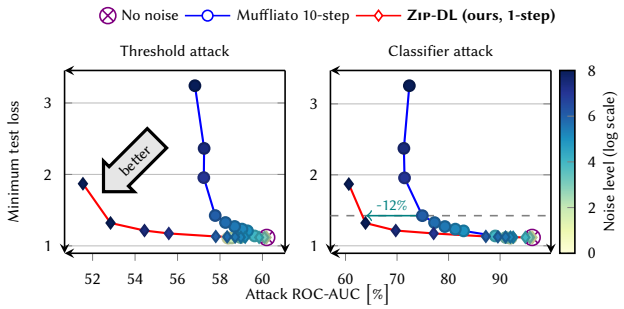

Figure 2: Minimum test loss as a function of attack results for varying noise levels on MovieLens. Color intensity represents a higher noise level. Zip-DL consistently shows better utility for equivalent attacker advantage.

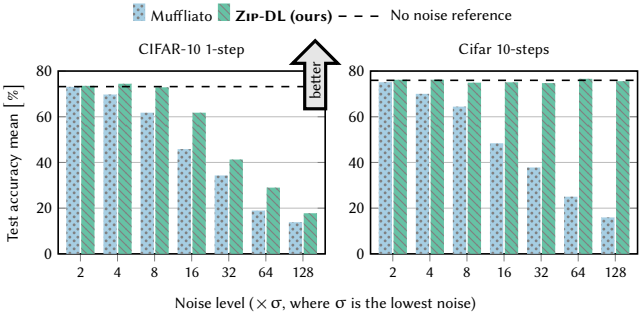

Figure 3: Best average test accuracy at different noise levels on CIFAR-10. Zip-DL is able to consistently reach higher accuracies, even for much higher noise levels.

case the attacker performs similarly to a random binary classifier. Better results are in the top-left corner (higher utility for a lower attack success). Figure 1 presents the results obtained for the two attacks in two communication settings: using 10 averaging steps (top row), and using 1 averaging step (bottom row). Results using the (basic) threshold attack are shown on the left, while those with the (stronger) classifier attack are on the right.

The charts show Zip-DL either matches or outperforms Muffliato in all four combinations. Zip-DL's advantage is notable at higher levels of protection when the ROC-AUC measure approaches 50%, where Zip-DL provides a better protection for the same utility, or a better utility for the same protection. This is particularly visible for 10 steps averaging (top row), where Zip-DL is able to neutralize the attack with close to no drop in accuracy. For instance, under the Threshold Attack, Zip-DL achieves a mean ROC AUC of 50.07% for an accuracy of 75.56%, a drop of only 0.25% percentage point against *No Noise* (75.81%). Zip-DL yields similar results under the Classifier Attack (50.42% ROC-AUC for the same test accuracy). This represents a substantial improvement over Muffliato which collapses at higher noise levels. For instance, under the Threshold Attack, Muffliato reaches a ROC-AUC of 50.13% at an accuracy of only 15.95%, representing a drop of 59.86% against Zip-DL (shown as a vertical cyan arrow on the figure.)

MovieLens. In contrast to Figure 1, Figure 2 captures model utility through average RMSE across nodes (test loss, vertical axis), where lower values are better. Figure 1 shows the results of the two attacks (Threshold Attack, left, and Classifier Attack, right) with different communication set-ups for the two competitors. While Muffliato uses 10 averaging steps (following the guidelines of [9]), Zip-DL only uses one. This allows Muffliato to converge to an almost exact average model across all 100 nodes between each learning iteration, while forcing Zip-DL to rely on imperfect averages limited to a node's immediate neighborhood. In spite of this disadvantage, Zip-DL clearly outperforms Muffliato on this task. Under the Threshold Attack, Zip-DL reaches a ROC-AUC of 52.84% for a test loss of 1.32 (second point from the left), a loss increase of only 0.21 against *No Noise* (ROC-AUC 60.20% for a test loss of 1.11), while Muffliato's ROC-AUC never gets below 56.535%, with a loss that diverges rapidly as noise increases. On the more powerful Classifier Attack, both Zip-DL and Muffliato grant a lesser protection. In spite of this, Zip-DL's protection advantage over its competitor is even higher, yielding a ROC-AUC of 63.87% at a loss of 1.32%, an improvement of close to 12% ROC-AUC points over the protection granted by Muffliato for a close to identical loss (ROC-AUC of 74.90% for a loss of 1.42, shown as an horizontal cyan arrow on the figure). For completeness, a figure detailing all averaging steps, as outlined in Figure 1, can be found in Appendix A.

Comparison. Comparing the results obtained on the two tasks, we observe clearly distinct behaviors:

- CIFAR-10: We observe in Figure 1 that even small perturbations can significantly reduce the attack efficiency. However, a higher noise level simply reduces the model's utility.
- MovieLens: By contrast, low noise levels have minimal influence on attack performance. But higher noise levels succeed in consistently dampening the attack's efficiency.

We conjecture that these different behaviors result from the fact that MovieLens considers a metric space in which the values of loss have a direct meaning for the attacker, while loss values are not directly significant in CIFAR-10. This is also the reason why we consider test accuracy on CIFAR-10 and the test loss on MovieLens. Regardless of these considerations, however, our results show that Zip-DL outperforms Muffliato in both cases.

Impact of the noise level. Figure 3 compares the best accuracy reached by Zip-DL and Muffliato for various noise levels on CIFAR-10. In contrast to Muffliato, the accuracy of Zip-DL is less sensitive to noise in the region of high test accuracy, *i.e.*, Zip-DL with a noise level of 8σ achieves similar test accuracy to Muffliato with a much lower noise level of 2σ. Furthermore, this remains true even when comparing Zip-DL with Muffliato 10-steps (even though Muffliato has a 10× communication cost). Interestingly, Zip-DL with 10 averaging steps does not experience any decrease in accuracy for all the noise levels we evaluated, since the noise cancellation can happen without proceeding through gradient descent. In conclusion, Zip-DL demonstrates better convergence when compared to Muffliato for similar privacy vulnerabilities, without requiring additional averaging steps.

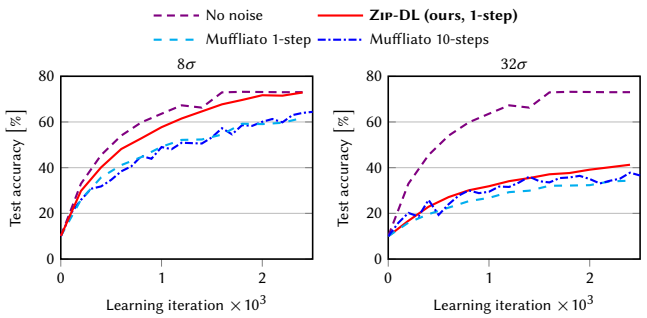

Figure 4: Muffliato test accuracy with different numbers of averaging rounds for a noise level of 8σ (left) and 32σ (right), compared to ZIP-DL (1-round). Even for high noise levels (32σ), Muffliato 10-steps marginally improves the performances over Muffliato 1-step, and ZIP-DL manages to beat both with only one averaging round.

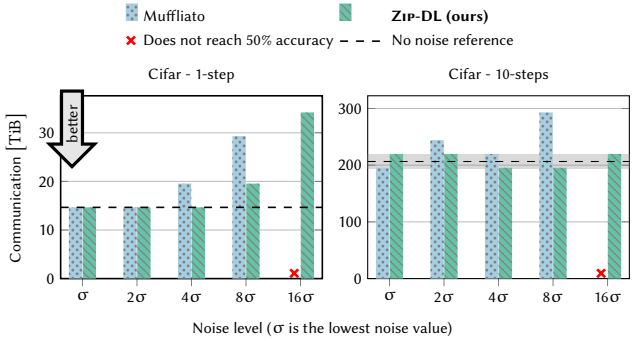

Figure 5: Communication cost to reach 50% accuracy for CIFAR-10. Muffliato fails to reach this target for higher noise levels.

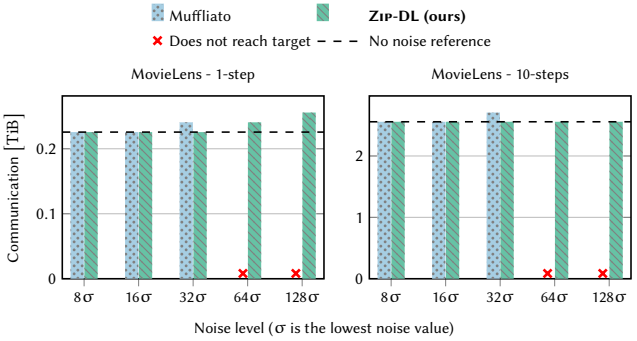

Figure 6: Communication cost to reach 1.25× the best test loss of unnoised DL for MovieLens. Muffliato fails to reach this target for higher noise levels.

6.3 Communication overhead

While basic DL and ZIP-DL limit themselves to a single averaging round per gradient descent step, Muffliato is designed to perform several of them to ensure the convergence of the averaging process. The exact number of communication steps required depends both on the variance of the models and datasets at the nodes and on the spectral analysis of the communication graph [9].

In our experimental scenario and with the same distribution assumptions as in Muffliato's original paper [9], we find that Muffliato's performs best with at least 10 averaging steps.

Figure 4 shows the evolution of the test accuracy w.r.t. the number of iterations for basic DL, Muffliato (with 1 and 10 averaging rounds), and ZIP-DL (with 1 averaging round) for two different noise levels. For both settings, we observe that Muffliato-10 is as accurate or more accurate than Muffliato-1. As evidenced by Figure 4, these additional averaging rounds have a heavier impact on the behavior of the test loss for higher noise levels, such as 32σ, but do not necessarily have much of an impact from a convergence point of view for lower noise levels, such as 8σ.

Our baseline [9] also considers Chebyshev polynomials for faster convergence. However, this only partially reduces the required averaging rounds and does not affect the results of Section 6.2 that only considers privacy and utility properties.

The addition of noise in both ZIP-DL and Muffliato does not affect only the final utility of the models. In some cases, this noise increases the number of learning iterations required for accuracy to converge, as the learning process must compensate for this noise, yielding more training time and communication overhead while also potentially exposing more information. We focus on this communication overhead and measure it using the total number of bytes transferred to reach 50% top-1 accuracy for both ZIP-DL and Muffliato. Figure 5 shows the communication overhead in TiB for increasing noise levels on the CIFAR-10 dataset. Being sensitive to the noise level, Muffliato does not even converge to an accuracy of 50% for noise levels beyond 16σ. ZIP-DL, therefore provides better privacy guarantees while limiting the communication overhead.

Finally, we also observe in Figure 5 that performing 10 averaging steps has a significant overhead when considering sizeable models, especially in combination with small local datasets that require frequent communication rounds between gradient descents. Here, running with 10 averaging steps yields an order of magnitude more communication cost to reach similar accuracy.

All those observations are reflected in Figure 6, which represents a comparable plot for the MovieLens dataset. Unlike the previous experiments, we focus here on the communication cost required to achieve a given target test loss rather than a target test accuracy, due to the nature of the MovieLens task. The target test loss is determined by adding 25% to the best unnoised test loss.

6.4 Other evaluation metrics

For completeness, we report in Figure 7 the privacy/utility tradeoff of ZIP-DL and Muffliato using the attacker's TPR at a low FPR on the MovieLens dataset. Intuitively, this represents the attacker's ability to identify valid training examples (TPR) when seeking to make close to no error (low FPR). We consider FPR values of both 0.1% and 1%, following [7]. For a FPR of 0.1%, the attack can be thwarted efficiently (down to a TPR of about 2.5%) with close to

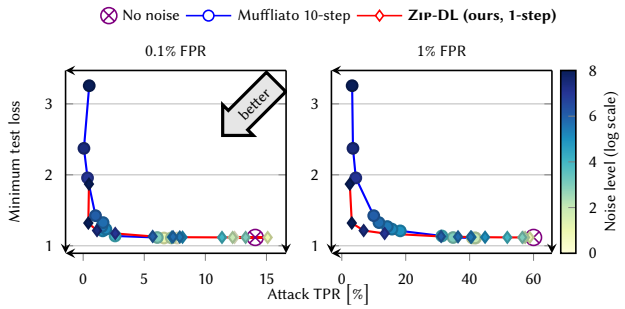

Figure 7: TPR at low FPR for the MovieLens dataset.

no impact on the test loss with both approaches, rendering a fined-grained comparison difficult. But for a FPR of 1%, Zip-DL provides a better tradeoff between test-loss and attack success, delivering better predictions for low TPR values (TPR $<$ 10%, bottom-left corner of the right-handed subfigure).

To further motivate our choice of attacks, we tried implementing a more recent MIA [7], but obtained almost worse results than random guesses in our DL scenarios. We conjecture this is due to both the non-IID nature of the data distribution in DL and the fact that DL usually considers small local training datasets. We believe these render this attack (and most shadow training attacks) impractical, and we view this as a promising avenue for future works.

7 Related work

Correlated noises. Correlated noises are a natural choice when seeking to reduce the utility cost of privacy. However, most of the literature focuses on correlation across nodes [2, 19, 35]. In order to apply such correlation, participating nodes need to rely either on a trusted aggregator, so that the noise can cancel out [19, 35], or on an agreement between nodes [2, 35]. We argue that the former is not always achievable, nor desirable, and the latter comes at a cost in terms of communication or utility [2].

A recent approach also leveraging correlated noises, DECOR [2], assumes that the channels between nodes may get compromised, and considers an adversary with access to every message transmitted on the network. Under this strong threat model, DECOR leverages shared secrets and introduces a novel privacy criterion, *secret-based local differential privacy* (SecLDP), which is orthogonal to PNDP considered in this paper. In particular, SecLDP is conditioned on the number of pairwise secrets compromised by the attacker. To counter such a strong adversary, DECOR injects a combination of independent and correlated noises that require pairwise coordination between nodes. This strategy is overkill in the presence of honest-but-curious adversaries and comes at a cost in terms of convergence and accuracy, a drawback Zip-DL does not exhibit (see Section 4.2).

Other methods to achieve privacy. Other variants of correlated noise, such as *secret sharing* [36] can be used in the context of DL [29]. While additive secret sharing does not necessarily involve coordination among nodes or a trusted aggregator, it requires multiple averaging rounds to reconstruct the shared average. Thus, additional operations, such as gradient descent, cannot be mixed together with the communication process, leading to prohibitive communication costs. Zip-DL, on the other hand is able to mix together communication and gradient computation (see Table 1).

Other cryptographic approaches include *secure multiparty computation* [21] and *secure aggregation* [5]. In these techniques, nodes agree on masks that conceal local models during the averaging process. Despite providing exact solutions to model averaging, they impose a significant drawback by requiring nodes to coordinate in order to set up and remove the masking. In large and dynamic distributed systems, this requirement may prove infeasible, especially in real-world scenarios involving mobile devices. For this reason, we designed our approach to avoid such coordination.

Combining with other privacy mechanisms. Because our approach requires no additional communication between nodes, it can also be used in combination with other approaches. For instance, uncorrelated noises could also be added to our approach. This would allow to have an intermediary approach, with possibly stronger privacy protection. Combining correlated and uncorrelated noises was proved to be possible [2] for other privacy definitions, the main difference with our approach being how the correlation is performed. Importantly, our approach does not necessitate any sort of coordination, making it friendly to combine with other works. However, this work focuses on fully correlated noises and their impact, and the study of such combinations is left for future work.

8 Conclusion

DL makes a step towards privacy in collaborative learning by preventing raw data sharing. However, models shared between nodes still leak private information. We introduce Zip-DL, which enhances privacy in DL by injecting correlated noise into shared models. Zip-DL does not introduce additional messages or require any sort of coordination across nodes, hence having minimal impact on communication cost while keeping convergence rates on par with the state-of-the-art. In particular, the noise introduced by Zip-DL has a provably lower impact on the convergence rate of the system than other similar approaches. Zip-DL can thus be used as a basic privacy addition even in high-performance regimes where traditional privacy-preserving mechanisms may be unusable because of utility degradation. We prove formal privacy guarantees for Zip-DL in terms of PNDP, bounding the privacy leakage of a node. Experimental results confirm Zip-DL's superior privacy-accuracy tradeoff under two paradigms of membership inference attacks with different levels of underlying strengths of the adversary's knowledge. Zip-DL performs particularly well on attacks that do not require crossing information across iterations, which are the most studied practical attack scenarios. Future work will explore broader scenarios beyond the initial assumptions of symmetric gossip matrices and behavior of a noisy gradient, aiming to extend Zip-DL's applicability and robustness guarantees.

Acknowledgments

Co-authors affiliated to EPFL have been funded by the Swiss National Science Foundation, under the project 'FRIDAY: Frugal, Privacy-Aware and Practical Decentralized Learning', SNSF proposal No. 10.001.796.

Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

This work was performed using HPC resources from GENCI-IDRIS (Grant 2024-AD011015352)

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*. Association for Computing Machinery, New York, NY, USA, 308–318. https://doi.org/10.1145/2976749.2978318

[2] Youssef Allouah, Anastasia Koloskova, Aymane El Firdoussi, Martin Jaggi, and Rachid Guerraoui. 2024. The Privacy Power of Correlated Noise in Decentralized Learning. https://doi.org/10.48550/arXiv.2405.01031 arXiv:2405.01031 [cs, math, stat]

[3] Enrique Tomás Martínez Beltrán, Mario Quiles Pérez, Pedro Miguel Sánchez Sánchez, Sergio López Bernal, Gérôme Bovet, Manuel Gil Pérez, Gregorio Martínez Pérez, and Alberto Huertas Celdrán. 2022. Decentralized federated learning: Fundamentals, state-of-the-art, frameworks, trends, and challenges. (2022). https://arxiv.org/abs/2211.08413

[4] Sayan Biswas, Mathieu Even, Anne-Marie Kermarrec, Laurent Massoulié, Rafael Pires, Rishi Sharma, and Martijn de Vos. 2025. Noiseless Privacy-Preserving Decentralized Learning. *Proceedings on Privacy Enhancing Technologies* 2025, 1 (Jan. 2025), 824–844. https://doi.org/10.56553/popets-2025-0043

[5] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1175–1191. https://doi.org/10.1145/3133956.3133982

[6] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. 2006. Randomized Gossip Algorithms. *IEEE Transactions on Information Theory* 52, 6 (June 2006), 2508–2530. https://doi.org/10.1109/TIT.2006.874516

[7] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. 2022. Membership Inference Attacks From First Principles. In *2022 IEEE Symposium on Security and Privacy (SP)*. 1897–1914. https://doi.org/10.1109/SP46214.2022.9833649

[8] Jin-Hua Chen, Min-Rong Chen, Guo-Qiang Zeng, and Jia-Si Weng. 2021. BDFL: A byzantine-fault-tolerance decentralized federated learning method for autonomous vehicle. *IEEE Transactions on Vehicular Technology* 70, 9 (2021), 8639–8652. https://doi.org/10.1109/TVT.2021.3102121

[9] Edwige Cyffers, Mathieu Even, Aurélien Bellet, and Laurent Massoulié. 2022. Muffliato: Peer-to-Peer Privacy Amplification for Decentralized Optimization and Averaging. In *Advances in Neural Information Processing Systems*.

[10] Martijn De Vos, Sadegh Farhadkhani, Rachid Guerraoui, Anne-marie Kermarrec, Rafael Pires, and Rishi Sharma. 2023. Epidemic Learning: Boosting Decentralized Learning with Randomized Communication. *Advances in Neural Information Processing Systems* 36 (Dec. 2023), 36132–36164.

[11] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. 2018. Minimax Optimal Procedures for Locally Private Estimation. *J. Amer. Statist. Assoc.* 113, 521 (Jan. 2018), 182–201. https://doi.org/10.1080/01621459.2017.1389735

[12] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography*. Springer Berlin Heidelberg, Berlin, Germany, 265–284. https://doi.org/10.1007/11681878_14

[13] Cynthia Dwork, Adam Smith, Thomas Steinke, and Jonathan Ullman. 2017. Exposed! A Survey of Attacks on Private Data. *Annual Review of Statistics and Its Application* 4, 1 (March 2017), 61–84. https://doi.org/10.1146/annurev-statistics-060116-054123

[14] Shripad Gade and Nitin H. Vaidya. 2018. Private Optimization on Networks. In *2018 Annual American Control Conference (ACC)*. IEEE, Milwaukee, WI, 1402–1409. https://doi.org/10.23919/ACC.2018.8430960

[15] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. 2020. Inverting Gradients - How Easy Is It to Break Privacy in Federated Learning?. In *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., 16937–16947.

[16] M. Gil, F. Alajaji, and T. Linder. 2013. Rényi Divergence Measures for Commonly Used Univariate Continuous Distributions. *Information Sciences* 249 (Nov. 2013), 124–131. https://doi.org/10.1016/j.ins.2013.06.018

[17] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems* 5, 4 (Jan. 2016), 1–19. https://doi.org/10.1145/2827872

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

[19] Hafiz Imtiaz, Jafar Mohammadi, Rogers Silva, Bradley Baker, Sergey M. Plis, Anand D. Sarwate, and Vince D. Calhoun. 2021. A Correlated Noise-Assisted Decentralized Differentially Private Estimation Protocol, and Its Application to fMRI Source Separation. *IEEE Transactions on Signal Processing* 69 (2021), 6355–6370. https://doi.org/10.1109/TSP.2021.3126546

[20] Márk Jelasity, Alberto Montresor, and Özalp Babaoglu. 2005. Gossip-based aggregation in large dynamic networks. *ACM Trans. Comput. Syst.* 23, 3 (2005), 219–252. https://doi.org/10.1145/1082469.1082470

[21] Renuga Kanagavelu, Qingsong Wei, Zengxiang Li, Haibin Zhang, Juniarto Samsudin, Yechao Yang, Rick Siow Mong Goh, and Shangguang Wang. 2022. CE-Fed: Communication efficient multi-party computation enabled federated learning. *Array* 15 (2022), 100207. https://doi.org/10.1016/j.array.2022.100207

[22] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2011. What Can We Learn Privately? *SIAM J. Comput.* 40, 3 (Jan. 2011), 793–826. https://doi.org/10.1137/090756090

[23] Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. 2020. A Unified Theory of Decentralized SGD with Changing Topology and Local Updates. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 5381–5393.

[24] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. 2014. The CIFAR-10 dataset. 55, 5 (2014). https://www.cs.toronto.edu/~kriz/cifar.html

[25] Batiste Le Bars, Aurélien Bellet, Marc Tommasi, Erick Lavoie, and Anne-Marie Kermarrec. 2023. Refined Convergence and Topology Learning for Decentralized SGD with Heterogeneous Data. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 206)*. PMLR, 1672–1702.

[26] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. 2017. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. In *NIPS*.

[27] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. 2018. Asynchronous Decentralized Parallel Stochastic Gradient Descent. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 3043–3052.

[28] Songtao Lu, Yawen Zhang, and Yunlong Wang. 2020. Decentralized federated learning for electronic health records. In *2020 54th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 1–5. https://doi.org/10.1109/CISS48834.2020.1570617414

[29] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS*.

[30] Ilya Mironov. 2017. Renyi Differential Privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. 263–275. https://doi.org/10.1109/CSF.2017.11 arXiv:1702.07476 [cs]

[31] Abdellah El Mrini, Edwige Cyffers, and Aurélien Bellet. 2024. Privacy Attacks in Decentralized Learning. https://doi.org/10.48550/arXiv.2402.10001 arXiv:2402.10001 [cs]

[32] Reza Nasirigerdeh, Javad Torkzadehmahani, Daniel Rueckert, and Georgios Kaissis. 2023. Kernel Normalized Convolutional Networks for Privacy-Preserving Machine Learning. In *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. 107–118. https://doi.org/10.1109/SaTML54575.2023.00016

[33] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning. In *2019 IEEE Symposium on Security and Privacy (SP)*. 739–753. https://doi.org/10.1109/SP.2019.00065 arXiv:1812.00910 [cs, stat]

[34] Róbert Ormándi, István Hegedüs, and Márk Jelasity. 2013. Gossip Learning with Linear Models on Fully Distributed Data. *Concurrency and Computation: Practice and Experience* 25, 4 (Feb. 2013), 556–571. https://doi.org/10.1002/cpe.2858 arXiv:1109.1396 [cs]

[35] César Sabater, Aurélien Bellet, and Jan Ramon. 2022. An Accurate, Scalable and Verifiable Protocol for Federated Differentially Private Averaging. *Machine Learning* 111, 11 (Nov. 2022), 4249–4293. https://doi.org/10.1007/s10994-022-06267-9

[36] Adi Shamir. 1979. How to Share a Secret. *Commun. ACM* 22, 11 (Nov. 1979), 612–613. https://doi.org/10.1145/359168.359176

[37] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*. IEEE Computer Society, 3–18. https://doi.org/10.1109/SP.2017.41

[38] Atul Singh, Tsuen-Wan Ngan, Peter Druschel, and Dan S. Wallach. 2006. Eclipse Attacks on Overlay Networks: Threats and Defenses. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications.* IEEE, Barcelona, Spain, 1–12. https://doi.org/10.1109/INFOCOM.2006.231

[39] Youliang Tian, Shuai Wang, Jinbo Xiong, Renwan Bi, Zhou Zhou, and Md Zakirul Alam Bhuiyan. 2023. Robust and privacy-preserving decentralized deep federated learning training: Focusing on digital healthcare applications. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (2023). https://doi.org/10.1109/TCBB.2023.3243932

[40] Yuxin Wu and Kaiming He. 2018. Group Normalization. In *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIII.* Springer-Verlag, Berlin, Heidelberg, 3–19. https://doi.org/10.1007/978-3-030-01261-8_1

[41] Lin Xiao, Stephen Boyd, and Sanjay Lall. 2006. Distributed Average Consensus with Time-Varying Metropolis Weights. *Automatica* (2006), 1–4.

[42] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M. Alvarez, Jan Kautz, and Pavlo Molchanov. 2021. See through Gradients: Image Batch Recovery via GradInversion. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE, Nashville, TN, USA, 16332–16341. https://doi.org/10.1109/CVPR46437.2021.01607

[43] Da Yu, Huishuai Zhang, Wei Chen, and Tie-Yan Liu. 2021. Do Not Let Privacy Overbill Utility: Gradient Embedding Perturbation for Private Learning. arXiv:2102.12677 [cs]

[44] Valentina Zantedeschi, Aurélien Bellet, and Marc Tommasi. 2020. Fully Decentralized Joint Learning of Personalized Models and Collaboration Graphs. In *AISTATS (Proceedings of Machine Learning Research, Vol. 108).* PMLR, 864–874.

[45] Oualid Zari, Chuan Xu, and Giovanni Neglia. 2021. Efficient Passive Membership Inference Attack in Federated Learning. *NeurIPS PriML 2021 - workshop Privacy in Machine Learning* (Oct. 2021). https://doi.org/10.48550/arXiv.2111.00430 arXiv:2111.00430

[46] Xinwei Zhang, Xiangyi Chen, Mingyi Hong, Steven Wu, and Jinfeng Yi. 2022. Understanding Clipping for Federated Learning: Convergence and Client-Level Differential Privacy. In *Proceedings of the 39th International Conference on Machine Learning.* PMLR, 26048–26067.

[47] Ligeng Zhu, Zhijian Liu, and Song Han. 2019. Deep Leakage from Gradients. In *Advances in Neural Information Processing Systems (NeurIPS)*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc.

A Additional experiments details

A.1 CIFAR-10: Zip-DL 1—step vs Muffliato 10—steps

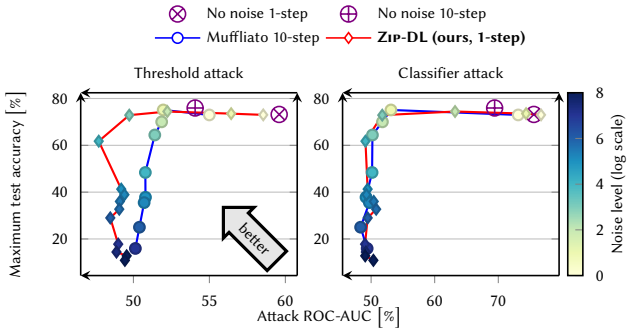

Figure 8: The privacy-utility tradeoff of Zip-DL compared to Muffliato 10-steps, on the CIFAR-10 dataset. Even though Zip-DL reach a similar tradeoff to Muffliato 10-steps, Zip-DL requires fewer communications.

As was done in Section 6.2 for MovieLens, we also directly compare Zip-DL to Muffliato with 10 averaging steps on CIFAR-10 in Figure 8. Looking at threshold attack, Zip-DL offers a better tradeoff. On the other hand, the tradeoffs of both approaches are very similar for the classifier attack. But closing this gap that is present in Figure 1 comes at the cost of 10 averaging steps, thus inducing a significant communication overhead. Thus, Zip-DL can achieve similar privacy properties compared to Muffliato 10-steps but reduces the communication overhead by a factor of 10.

A.2 MovieLens: Zip-DL 1—step vs Muffliato 10—steps

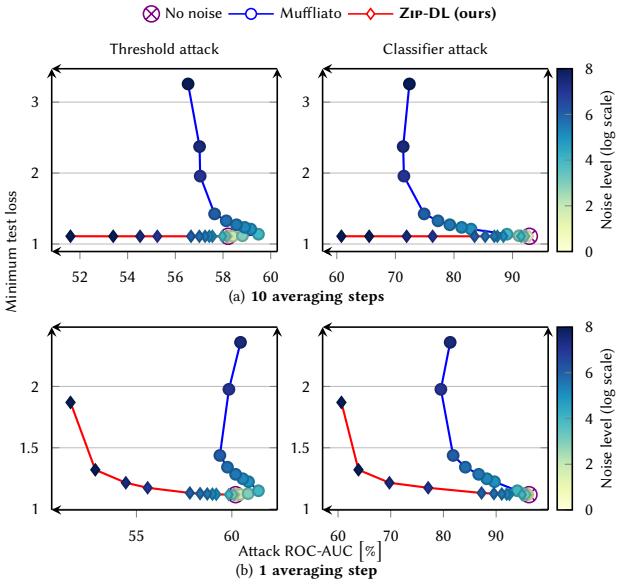

Figure 9: The privacy-utility tradeoff of Zip-DL compared to Muffliato on MovieLens, for multiple averaging steps.

We also display the full tradeoffs for MovieLens in Figure 9, mimicking what was done for CIFAR-10 in Figure 1. We observe a similar tendency: our approach systematically offers a better tradeoff. Moreover, the tradeoff offered by Muffliato 1–steps is significantly worse compared to Muffliato 10–steps. This further motivates our choice for Figure 2 to directly compare Zip-DL with 1 averaging step versus Muffliato with 10 averaging steps.

A.3 Clipping

In practice, Assumption 5.1 can be enforced through gradient clipping, a standard approach in DP-SGD [1, 2]. However, gradient clipping introduces additional constraints on theoretical convergence, and in some scenarios, it may even prevent convergence entirely [46].

To assess the impact of clipping on the convergence guarantees of Zip-DL, we evaluate its effect on convergence in Figure 10. Our experiments focus on the MovieLens dataset with two clipping parameters: $\Delta = 1$ and $\Delta = 0.5$.

Figure 10 shows that the convergence of Zip-DL is only marginally affected by the clipping parameter, even when the noise variance approaches the gradient bound. This illustrates that Assumption 5.1 can be effectively enforced in practice through gradient clipping without significantly compromising convergence.

Table 2: List of the main symbols used in this work.

Symbol	Usage
$\mathcal{V}$	Set of all the nodes that participate in the training.
n	Number of nodes in $\mathcal{V}$.
a, u, v	Nodes in $\mathcal{V}$.
$\Gamma_a^{(t,s)}$	Neighbors of node a at averaging round s, after learning iteration t.
$d_a^{(t,s)}$	Degree of node a at averaging round s, after learning iteration t.
d_a	Maximum degree of node a, over learning iterations and averaging rounds.
$W^{(t,s)}$	Gossip matrix at averaging round s, after learning iteration t.
p	Mixing parameter of the gossip matrices (Assumption 2.1).
$x_a^{(t)}$	Model of node a at learning iteration t.
$\bar{x}^{(t)}$	Average model at learning iteration t.
$x_a^{(t+1/2)}$	Model of node a at learning iteration t after the gradient step.
$\bar{x}^{(t+1/2)}$	Average model at learning iteration t after the gradient step.
x^*	Optimal model.
f^*	Minimum of the global loss function.
$\mathcal{D}_a$	Data distribution of node a.
$\xi_a^{(t)}$	Data sample drawn from $\mathcal{D}_a$.
F_a	Loss function of node a.
f_a	Sampled (or expected) loss of node a (Equation (1)).
f	Globally sampled loss (Equation (1)).
μ	Convexity constant (Assumption 4.1).
L	Smoothness constant (Assumption 4.1).
ϑ_i^2	Noise level at the optimum (Assumption 4.3).
ω_i^2	Diversity of the data distribution at the optimum (Assumption 4.3).
γ	Stepsize of the gradient descent.
$Y_{a\to v}^{(t)}$	Intermediate noise generated by node a destined to v at learning iteration t.
$Z_{a\to v}^{(t)}$	Zip-DL-averaging noise from node a to node v at learning iteration t.
ς_a^2	Variance of $Y_{a\to v}^{(t)}$.
$(\sigma_{a\to v}^{(t)})^2$	Variance of $Z_{a\to v}^{(t)}$.
Δ	Adjacent datasets bound (Assumption 5.1).
$g^{(T)}(a, v)$	Privacy bound from node a to node v at timestamp T (Definition 5.3).
$\hat{X}^{(t)}$	Virtual models vector at time t.
$\hat{\chi}^{(t)}$	Unnoised virtual execution (with the same graphs and batches, but no noise) at time t.
$\hat{M}$	(Virtual) Mixing matrix (Equation (8))
$^{\intercal}A$	Transpose of some matrix A.

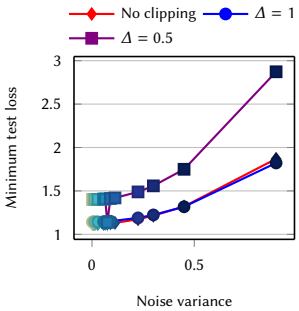

Figure 10: Best test loss of Zip-DL for different clipping parameters, on the MovieLens dataset.

A.4 Experiments with varying number of nodes

For completeness, we also explore varying the number of nodes partaking in DL, which was fixed to 100 in the main paper. We vary this count to respectively 32 and 64 nodes in Figure 11.

We observe the tradeoffs keep the same tendency as the ones of Figure 2. Moreover, the order of magnitude of this tradeoff is consistent, even if the number of nodes and the degree of the graph was changed.

A.5 Node dropout

Since Zip-DL relies on correlated noise sent in different directions through the network, it is interesting to evaluate how node dropout impacts Zip-DL's performances. Such a scenario is not computationally expensive: since each node behaves independently, a node dropout does not introduce any need for restarting a round. This is

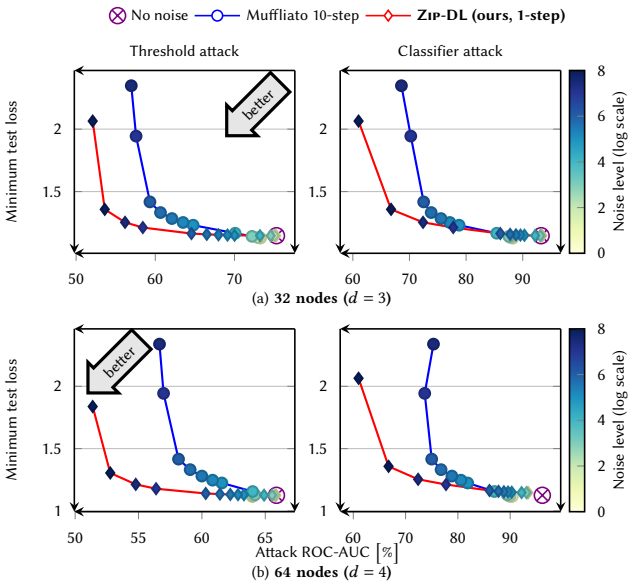

Figure 11: Varying number of nodes for the MovieLens dataset.

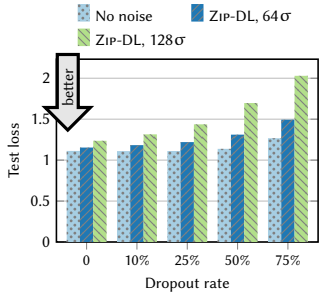

Figure 12: Test loss for a fixed communication budget with varying dropout rates for the MovieLens dataset.

one of the benefits of our approach compared to secret sharing since we add noise and not masks. However, node dropout will naturally impact any DL approach. In our case, it may change Lemma 3.1, since the propagated noise will not be exactly zero. We evaluate in this section how node dropout affects Zip-DL.

We call node dropout when a node skips a communication round. Such a node does not compute any gradient and does not receive or send any message. However, it may come back online later in the training. To do so, we simulate four levels of dropout: 10%, 25%, 50% and 75%. We also add a dropout correlation of 10%, making the dropped out nodes more likely to remain dropped out.

Figure 12 reports the result, by considering the test loss for two noise levels and those for dropout rates. We observe that dropout makes higher noise level deteriorate more in test loss. However, even for high levels of dropout and noise, the degradation remains marginal as long as the dropout rate is below 25%.

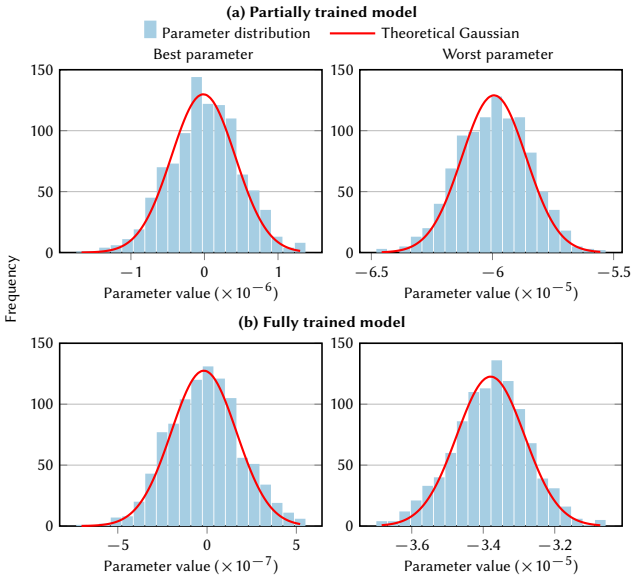

Figure 13: Distribution of the noisy gradient for two parameters on the MovieLens dataset, with low noise level σ

This decrease in utility for high levels of dropout is to be expected, as this is outside the considerations of our theoretical convergence guarantee. In particular, node dropout makes Lemma 3.1 not hold, which was pivotal in our proof. However, we conjecture that our convergence proof could be adapted to capture node dropouts, at the cost of a term in the form of σ^2/T. This would mean our convergence mostly matches the existing literature [2, 9] in terms of noise impact. However, how the dropout rate will affect the noise term in $1/T$ remains a future work, as our approach is initially designed for networks with a low amount of dropout to leverage the noise cancellation.

A.6 Empirical evaluation of Assumption 5.6

We empirically examine some simple scenarios to provide motivation for and, in turn, justify Assumption 5.6.

To achieve this, we utilize a centralized MovieLens task with the same parameters as those described in Section 6.1. We consider two cases: a fully trained model (after 100 iterations) and a partially trained model (after 10 iterations). For each case, we examine two data samples of size 1000 each. In the first sample, we compute the true gradient and subsequently generate a noisy gradient by perturbing this true gradient. In the second sample, we first add noise to the model parameters and then compute the gradient. We evaluate the differences between the two normalized distributions by calculating the Kolmogorov-Smirnov test statistic for all parameters, given that the exact Lipschitz constant L is unknown.

For clarity, we present two histograms illustrating the distribution in the scenario where we first add noise and then compute the gradient: one for the best-case and the other for the worst-case Kolmogorov-Smirnov test outcomes. To optimize computational

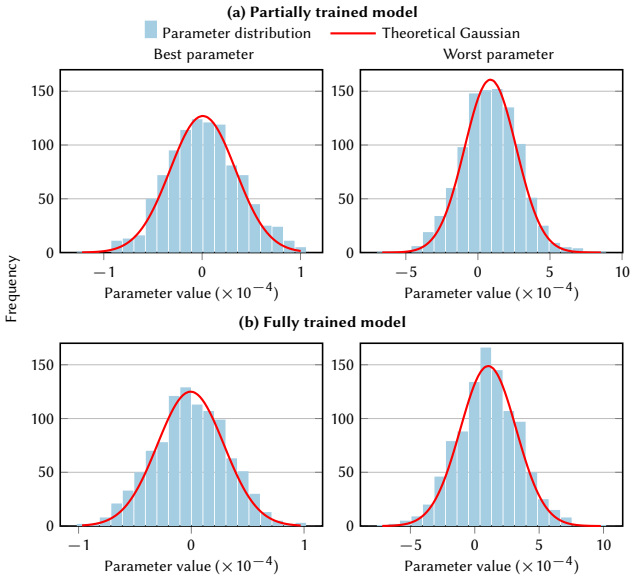

Figure 14: Distribution of the noisy gradient for two parameters on the MovieLens dataset, with high noise level 128σ

efficiency, we randomly select 10,000 parameters for calculating the aforementioned metrics.

Figure 13 shows the results for a low noise level (σ), while Figure 14 illustrates the outcomes for a higher noise level (128σ). More specifically, Figures 13 and 14, subplots (a), presents the case with a partially trained model, whereas Figures 13 and 14, subplots (b), represents gradients computed on a noise around the fully trained model. Across all scenarios, a consistent trend is observed for all parameters, indicating that the parameters generally follow a normal distribution.

A.7 Additional attacks details

In this section, we provide further details to the MIA workflow for both the threshold and the classifier attack. We also specify what information the attacker has access to.

Attacker observations during training. During training, an attacker stores models received from its neighbors at different iterations, while denoting whose model is being saved. It is important to note those are noisy models. For CIFAR-10, the attacker stores one every 100 iterations, whereas it is one every 50 iteration for MovieLens.

Threshold attack. The attacker computes the losses generated by the model on both the victim's local training set and the global test set. The attacker evaluates the ROC-AUC for each saved model. This yields an attack result for each logged model, meaning we can also observe tendencies across iterations.

Classifier attack. After training, the attacker groups all the models received that were sent by a target victim node v. For a given data point x, the attacker computes the loss of x through all logged models of v. This creates a time series of losses for this data point x. We can label those time series considering whether $x \in \mathcal{D}_v$ or not. The attacker then trains a classifier to discriminate between time series, using a train-test split between both the testing set and the victim's local training set. 70% of the local training dataset and the testing set are used to train the classifier, and the remaining 30% of both sets are used for evaluation. Reweighting is performed to account for the unbalance in the class distributions.

B Privacy proof

In this section, we provide the necessary steps to prove Theorem 5.9. Most necessary assumptions are detailed in Sections 4.1 and 5.1, but Section B.1 details common technical assumptions in the field, or express intermediary results necessary for the main result. Then, Section B.2 proves the main privacy theorem. Finally, Section B.3 contains details about Section 5.5.

B.1 Assumptions and lemmas

We start by proving the equivalent system matrix formulation.

Lemma 5.7. *Consider $i \in \mathcal{V}$ and $t \in \mathbb{N}$. Then we have:*

$$\forall k \in \mathcal{V}, \hat{X}^{(t)}_{ni+k} = X^{(t)}_i.$$

Proof. (Lemma 5.7) We proceed by induction over $t \in \mathbb{N}$, using (9) and unrolling the matrix multiplication. The initialization is done by definition. Now, assume that $\hat{X}^{(t)}_{ni+k} = X^{(t)}_i$ for all k.

First, we observe that we have $\hat{X}^{(t+1/2)}_{ni+k} = X^{(t+1/2)}_i$ by definition. Second, we have, using (9):

$$\begin{aligned}\hat{X}^{(t+1)}_{ni+k} &= \left(\hat{M}\hat{W}^{(t)}(\hat{X}^{(t+1/2)} + \hat{Z}^{(t)})\right)_{ni+k} \\ &= \sum_{\hat{j}=0}^{n^2} \hat{M}_{ni+k,\hat{j}} \left(\hat{W}^{(t)}(\hat{X}^{(t+1/2)} + \hat{Z}^{(t)})\right)_{\hat{j}}\end{aligned}$$

We can remove indexes in the virtual domain by exploiting the following properties of the matrices:

- $\hat{M}_{n(i-1)+k,\hat{j}} \neq 0 \iff \hat{j} \in [[n(i-1)+1, ni]]$ and $\hat{M}_{n(i-1)+k,\hat{j}} = 1$ in this case in such case, which simplifies the sum by removing $\hat{M}$.
- $\hat{W}^{(t)}_{j+n(i-1),\hat{u}} \neq 0 \iff \hat{u} = n(j-1)+i$, in which case $\hat{W}^{(t)}_{j+n(i-1),\hat{u}} = W^{(t)}_{i,j}$, which simplifies the sum further, by removing indexes and rewriting in terms of $W^{(t)}$.

Thus, we get:

$$\begin{aligned}\hat{X}_{ni+k}^{(t+1)} &= \sum_{\hat{j}=n(i-1)+1}^{ni} \left(\hat{W}^{(t)}(\hat{X}^{(t+1/2)} + \hat{Z}^{(t)})\right)_{\hat{j}} \\ &= \sum_{j=1}^{n} \left(\hat{W}^{(t)}(\hat{X}^{(t+1/2)} + \hat{Z}^{(t)})\right)_{n(i-1)+j} \\ &= \sum_{j=1}^{n} \sum_{\hat{u}=0}^{n^2} \hat{W}^{(t)}_{n(i-1)+j,u} \left(\hat{X}^{(t+1/2)} + \hat{Z}^{(t)}\right)_{\hat{u}} \\ &= \sum_{j=1}^{n} W_{i,j}^{(t)} \left(\hat{X}^{(t+1/2)} + \hat{Z}^{(t)}\right)_{n(j-1)+i} \\ &= \sum_{j=1}^{n} W_{i,j}^{(t)} \left(X_j^{(t+1/2)} + Z_{j\to i}^{(t)}\right)\end{aligned}$$

Where we used the induction hypothesis. Now, we use the observation that $\hat{X}_{ni+k}^{(t+1/2)} = X_i^{(t+1/2)}$ to conclude our induction:

$$\begin{aligned}\hat{X}_{ni+k}^{(t+1)} &= \sum_{j=1}^{n} W_{i,j}^{(t)} \left(X_j^{(t+1/2)} + Z_{j\to i}^{(t)}\right) \\ &= X_i^{(t+1)}.\end{aligned}$$

□

Lemma 5.8. *Using Assumption 5.6, consider $\hat{\chi}^{(T)}$ a virtual execution without any noise, and every other source of randomness is the same. Then, we have:*

$$\hat{X}^{(T)} \sim \mathcal{N}\left(\hat{\chi}^{(T)}, L\hat{W}^{(T)}\tilde{C}^{(t)}\Sigma_{\tilde{Y}^{(t)}}{}^{\intercal}(\hat{W}^{(T)}\tilde{C}^{(t)})\right).$$

Here, $\Sigma_Z^{(t)}$ represents correlated noises that will cancel out, for t big enough we have $(\hat{M}\hat{W})^t\Sigma_Z{}^{\intercal}(\hat{M}\hat{W})^t = 0$. Thus, once T is big enough, the variance $\Sigma_Z^{(t)}$ will become constant.

Proof. (Lemma 5.8) We proceed by induction on T for the expected value, and note $\Sigma_T = \sum_{t=1}^{T}(1-\gamma L)^t(\hat{M}\hat{W})^t\Sigma_Z^{(t)\intercal}(\hat{M}\hat{W})^t$. We have the following two update rules:

$$\begin{aligned}\hat{X}^{(T+1)} &= \hat{M}\hat{W}^{(T)}\left(\hat{X}^{(T)} - \gamma\nabla F(\hat{X}^{(T)}, \xi^{(T)}) + \hat{Z}^{(t)}\right) \\ \hat{\chi}^{(T+1)} &= \hat{M}\hat{W}^{(T)}\left(\hat{\chi}^{(T)} - \gamma\nabla F(\hat{\chi}^{(T)}, \xi^{(T)})\right).\end{aligned}$$

First, we can show by another induction that this is a linear combination of Gaussian random variables.

Then, let us look at the expected value for $\hat{X}^{(T+1)}$: if we assume that the expected value of $\hat{X}^{(T)}$ is $\hat{\chi}^{(T)}$, Assumption 5.6 guarantees that the expected value of $\hat{X}^{(T+1)}$ is $\hat{\chi}^{(T+1)}$.

Finally, using Assumption 5.6, we have:

$$\hat{X}^{(T+1)} \sim \mathcal{N}\left(\hat{\chi}^{(T+1)}, L\Sigma_{T+1}\right)$$

With the following update rule:

$$\begin{aligned}L\Sigma_{T+1} = (1-\gamma L)\hat{M}\hat{W}^{(T)}(L\Sigma_T)^{\intercal}(\hat{M}\hat{W}^{(T)}) \\ + L(\hat{M}\hat{W}^{(T)})\Sigma_Z{}^{\intercal}(\hat{M}\hat{W}^{(T)}).\end{aligned}$$

This yields the following:

$$\begin{aligned}L\Sigma_T &= L\sum_{t=1}^{T}(1-\gamma L)^t(\hat{M}\hat{W})^t\Sigma_Z^{(t)\intercal}(\hat{M}\hat{W})^t \\ &= L\tilde{W}^{(T)}\Sigma_Z^{(t)\intercal}\tilde{W}^{(T)}.\end{aligned}$$

□

To prove PNDP, we will need a bound between two adjacent inputs is derived using the following lemma:

Lemma B.1. *Consider two unnoised executions. Then,*

$$\left\|\hat{\chi}^{(t)} - \hat{\dot{\chi}}^{(t)}\right\|_\infty^2 \leq \frac{4\gamma^2\Delta^2}{1+4\gamma^2 L}((2+4\gamma^2 L)^t - 1).$$

This lemma bounds the maximal difference between local models of two adjacent unnoised executions. One limitation of this lemma is that it bounds over a maximum. This is because a gradient term must be isolated from the recursive term in the proof. To show this, Section 5.5 focuses on the case where only averaging is performed, and no gradient descent. In this scenario, the equivalent of the above lemma is tighter, and we derive a generalization of previous results to our case of correlated noises.

Proof. (Lemma B.1) We know that $\left\|\hat{M}\hat{W}\right\|_\infty = 1$.

$$\begin{aligned}\left\|\hat{\chi}^{(t)} - \hat{\dot{\chi}}^{(t)}\right\|_\infty^2 &\leq \left\|\hat{M}\hat{W}^{(t)}\left(\hat{\chi}^{(t-1/2)} - \hat{\dot{\chi}}^{(t-1/2)}\right)\right\|_\infty^2 \\ &\leq 2\left\|\hat{\chi}^{(t-1)} - \hat{\dot{\chi}}^{(t-1)}\right\|_\infty^2 + 2\gamma^2 C_1^{(t)},\end{aligned}$$

With $C_1^{(t)} := \left\|\nabla F(\hat{\chi}^{(t-1)}, \xi^{(t-1)}) - \nabla F(\hat{\dot{\chi}}^{(t-1)}, \dot{\xi}^{(t-1)})\right\|_\infty^2$.

We focus on the left term, and notice that:

$$\begin{aligned}C_1^{(t)} &\leq 2\left\|\nabla F(\hat{\chi}^{(t-1)}, \xi^{(t-1)}) - \nabla F(\hat{\chi}^{(t-1)}, \dot{\xi}^{(t-1)})\right\|_\infty^2 \\ &\quad + 2\left\|\nabla F(\hat{\chi}^{(t-1)}, \dot{\xi}^{(t-1)}) - \nabla F(\hat{\dot{\chi}}^{(t-1)}, \dot{\xi}^{(t-1)})\right\|_\infty^2 \\ &\overset{(5),(2)}{\leq} 2\Delta^2 + 2L\left\|\hat{\chi}^{(t-1)} - \hat{\dot{\chi}}^{(t-1)}\right\|_\infty^2\end{aligned}$$

Thus, we get:

$$\left\|\hat{\chi}^{(t)} - \hat{\dot{\chi}}^{(t)}\right\|_\infty^2 \leq (2+4\gamma^2 L)\left\|\hat{\chi}^{(t-1)} - \hat{\dot{\chi}}^{(t-1)}\right\|_\infty^2 + 4\gamma^2\Delta^2$$

Unrolling the recursion, we obtain:

$$\left\|\hat{\chi}^{(t)} - \hat{\dot{\chi}}^{(t)}\right\|_\infty^2 \leq \frac{4\gamma^2\Delta^2}{1+4\gamma^2 L}((2+4\gamma^2 L)^t - 1).$$

□

B.2 Proof of the main theorem

In this section, we remind and provide a full proof of Theorem 5.9.

Theorem 5.9 (Privacy of Zip-DL). *T iterations of Zip-DL (Algorithm 2) satisfies $\left(\alpha, \epsilon^{(T)}(a,v)\right)$-PNDP, where $\epsilon^{(T)}(a,v)$ is bounded for any two nodes $a, v \in \mathcal{V}$ by:*

$$\frac{2\alpha\gamma^2\Delta^2}{L+4\gamma^2L^2}\sum_{t=0}^{T-1}\sum_{\substack{\hat{v}\in\hat{V}\\ \hat{w}\in\hat{\Gamma}_{\hat{v}}^{(t)}}}\frac{(2+4\gamma^2L)^t - 1}{\left(\left(\tilde{W}\tilde{C}\right)^{(t)}\tilde{\Sigma}_{\tilde{Y}^{(t)}}{}^{\intercal}\left(\tilde{W}\tilde{C}\right)^{(t)}\right)_{\tilde{w},\tilde{w}}},$$

where $\tilde{\Sigma}_{\tilde{Y}^{(t)}}$ is a diagonal matrix representing the noise variances of all noises generated by the algorithm up to time T, $\tilde{C}^{(t)}$ is a block-diagonal matrix representing the correlation factor at each iteration t, and $\tilde{W}^{(t)}$ is the accumulation of all the powers of the gossip matrix defined in Section 5.2.

Proof. We want to bound the privacy loss that emerges from the view of nodes V. To this end, we will use the matrix notations defined in Section 5.2, with a virtual network.

For simplicity of notation, we assume that the communication matrix is fixed through time. The proof generalizes to arbitrary communication matrix at time t at the expense of product notations. We obtain the following update rule for a given averaging round t:

$$\hat{X}^{(t+1)} = \hat{M}\hat{W}\left(\hat{X}^{(t)} - \gamma\nabla F(\hat{X}^{(t)}, \xi^{(t)}) + \hat{Z}^{(t)}\right) \tag{12}$$

We now want to focus on two distinct executions on datasets $\xi^{(t)} \sim_u \dot{\xi}^{(t)}$. The dot notation will correspond to the execution of the algorithm on an adjacent dataset.

If we now consider some set of nodes $V \subseteq \mathcal{V}$, we denote $\hat{V} \subseteq \hat{\mathcal{V}}$ the set of corresponding virtual nodes. We name P_V^T the privacy loss:

$$P_V^{(T)} := D_\alpha\left(\mathcal{O}_{\hat{V}}(\mathcal{A}^{(T)}(\mathcal{D}))\|\mathcal{O}_{\hat{V}}(\mathcal{A}^{(T)}(\dot{\mathcal{D}}))\right).$$

We want to bound:

$$\begin{aligned} P_V^{(T)} &= D_\alpha\left(\mathcal{O}_{\hat{V}}(\mathcal{A}^{(T)}(\mathcal{D}))\|\mathcal{O}_{\hat{V}}(\mathcal{A}^{(T)}(\dot{\mathcal{D}}))\right) \\ &\leq \sum_{t=0}^{T-1}\sum_{\hat{v}\in\hat{V}}\sum_{\hat{w}\in\hat{\Gamma}_{\hat{v}}^{(t)}} D_\alpha\left(\hat{X}_{\hat{w}}^{(t)}\|\hat{\dot{X}}_{\hat{w}}^{(t)}\right) \end{aligned} \tag{13}$$

Our main focus is thus to bound $D_\alpha\left(\hat{X}_{\hat{w}}^{(t)}\|\hat{\dot{X}}_{\hat{w}}^{(t)}\right)$. To this end, we want to apply Lemma 5.8 to both $\hat{X}_{\hat{w}}^{(t)}$ and $\hat{\dot{X}}_{\hat{w}}^{(t)}$. One key remark is that both their distributions are centered on slightly altered trajectories, corresponding to the two adjacent datasets. Thus, we apply Lemma 5.8, and obtain:

$$\hat{X}_{\hat{w}}^{(t)} \sim \mathcal{N}(\hat{\chi}_{\hat{w}}^{(t)}, L(\Sigma_T)_{\hat{w},\hat{w}}), \qquad \hat{\dot{X}}_{\hat{w}}^{(t)} \sim \mathcal{N}(\hat{\dot{\chi}}_{\hat{w}}^{(t)}, L(\Sigma_T)_{\hat{w},\hat{w}}),$$

with $\Sigma_T = \sum_{t=1}^{T}(1-\gamma L)^t(\hat{M}\hat{W})^t\Sigma_Z{}^\intercal(\hat{M}\hat{W})^t$.

One last thing we may want to do is factorize the noise expression: We now consider the matrix of all the noises $\tilde{Z}^{(T)} \in \mathbb{R}^{Tn^2}$, where $\tilde{Z}[tn^2 + \hat{w}] := \hat{Z}_{\hat{w}}^{(t)}$ for $0 \leq \hat{w} < n^2$. We can express the term by considering the temporal matrix notations of Section 5.2. This leads to:

$$\Sigma_T = \tilde{W}^{(T)}\tilde{M}\Sigma_{\tilde{Y}}{}^\intercal(\tilde{W}^{(T)}\tilde{M}) \tag{14}$$

Considering (11),(14) along with Lemma 5.4, we obtain:

$$D_\alpha\left(\hat{X}_{\hat{w}}^{(t)}\|\hat{\dot{X}}_{\hat{w}}^{(t)}\right) \leq \frac{\alpha}{2L}\frac{\left\|\hat{\chi}_{\hat{w}}^{(t)} - \hat{\dot{\chi}}_{\hat{w}}^{(t)}\right\|^2}{\left(\tilde{W}^{(t)}\tilde{M}\Sigma_{\tilde{Y}}{}^\intercal(\tilde{W}^{(t)}\tilde{M})\right)_{\tilde{w},\tilde{w}}}$$

Finally, we need to bound the difference between the two unnoised executions $\left\|\hat{\chi}_{\hat{w}}^{(t)} - \hat{\dot{\chi}}_{\hat{w}}^{(t)}\right\|^2$ using Lemma B.1.

Putting it all together in (13), we can bound:

$$P_V^{(T)} \leq \frac{2\alpha\gamma^2\Delta^2}{L+4\gamma^2L^2}\sum_{t=0}^{T-1}\sum_{\hat{v}\in\hat{V}}\sum_{\hat{w}\in\hat{\Gamma}_{\hat{v}}^{(t)}}\frac{(2+4\gamma^2L)^t - 1}{\left(\tilde{W}^{(t)}\tilde{M}\Sigma_{\tilde{Y}}{}^\intercal(\tilde{W}^{(t)}\tilde{M})\right)_{\tilde{w},\tilde{w}}}. \tag{15}$$

□

B.3 Proof of the averaging algorithm

We prove Section 5.5.

Theorem 5.12. *T iterations of Algorithm 1 satisfy $(\alpha, \epsilon^{(T)}(a,v))$-PNDP, where $\epsilon^{(T)}(a,v)$ is bounded for any two nodes $a, v \in \mathcal{V}$ by:*

$$\frac{\alpha\Delta^2}{2}\sum_{t=0}^{T-1}\sum_{\hat{v}\in\hat{V}}\sum_{\hat{w}\in\hat{\Gamma}_{\hat{v}}^{(t)}}\frac{\left((\hat{M}\hat{W})^T\right)_{\hat{w},\hat{a}}}{\left((\tilde{W}\tilde{C})^{(t)}\Sigma_{\tilde{Y}}{}^\intercal(\tilde{W}\tilde{C})^{(t)}\right)_{\tilde{w},\tilde{w}}},$$

where

$$\tilde{W}^{(T)} := \left((\hat{M}\hat{W})^T, \quad \ldots, \quad \hat{M}\hat{W}\right).$$

Sketch of proof. (Theorem 5.12) We can follow the same proof concept for the averaging algorithm presented in Algorithm 1. In this case, the notion of adjacent dataset is slightly different, as it concerns the original data itself $X^{(0)}$. We will obtain a simpler update rule:

$$\hat{X}^{(T+1)} = \hat{M}\hat{W}^{(T)}\left(\hat{X}^{(T)} + \hat{Z}^{(T)}\right).$$

Unrolling the model updates, and following a similar reasoning, we obtain that:

$$\begin{aligned} \hat{X}^{(T+1)} &\sim \mathcal{N}((\hat{M}\hat{W})^T\hat{X}^{(0)}, \tilde{W}^T\tilde{M}\Sigma_Y{}^\intercal(\tilde{W}^T\tilde{M})) \\ \text{where } \tilde{W}^{(T)} &:= \left((\hat{M}\hat{W})^T, \quad \ldots, \quad \hat{M}\hat{W}\right) \in \mathbb{R}^{n^2\times Tn^2} \end{aligned}$$

Then, using the same decomposition and Lemma 5.4, we observe the sensitivity is:

$$\left\|\left((\hat{M}\hat{W})^T\left(\hat{X}^{(0)} - \hat{\dot{X}}^{(0)}\right)\right)_{\hat{w}}\right\|^2 \leq \left((\hat{M}\hat{W})^T\right)_{\hat{w},\hat{u}}\Delta^2,$$

with Δ the bound on two adjacent datasets, since $\hat{X}^{(0)}$ and $\hat{\dot{X}}^{(0)}$ are only different in component u. We can derive the desired result from this. □

C Proofs of Zip-DL main properties

This section contains proofs to Section 3.

Lemma 3.1. Noise cancellation on the global model. *For every node $a \in \mathcal{V} = [\![1,n]\!]$, it holds that*

$$\sum_{v=1}^{n} W_{a,v}Z_{a\to v} = 0 = \sum_{v=1}^{n} W_{v,a}Z_{a\to v}.$$

Proof. Using the notation in Algorithm 1, and since the matrix is symmetric, we have for a fixed node a:

$$\begin{aligned}\sum_{v\in\Gamma_a} W_{a,v}Z_{a\to v} &= \sum_{v\in\Gamma_a} W_{a,v}[Y_{a\to v} - \frac{1}{d_a W_{a,v}}\sum_{j\in\Gamma_a} W_{a,j}Y_{a\to j}]\\ &= \sum_{v\in\Gamma_a} W_{a,v}Y_{a\to v} - \sum_{v\in\Gamma_a}\frac{1}{d_a}(\sum_{j\in\Gamma_a} W_{a,j}Y_{a\to j})\\ &= \sum_{v\in\Gamma_a} W_{a,v}Y_{a\to v} - \sum_{j\in\Gamma_a} W_{a,j}Y_{a\to j}\\ &= 0.\end{aligned}$$

□

Corollary 3.2. Impact on the global average model. *For every epoch $t \in [[0,T]]$, we have:*

$$\bar{x}^{(t+1)} = \bar{x}^{(t+1/2)}.$$

Proof.

$$\begin{aligned}\bar{x}^{(t+1)} &= \frac{1}{n}\sum_{a=1}^n x_a^{(t+1)} = \frac{1}{n}\sum_{a=1}^n\sum_{v\in\Gamma_a} W_{a,v}^{(t)}(x_v^{(t+1/2)} + Z_{v\to a}^{(t)})\\ &= \frac{1}{n}\sum_{a=1}^n\sum_{v\in\Gamma_a} W_{a,v}^{(t)}x_v^{(t+1/2)} + \frac{1}{n}\sum_{a=1}^n\sum_{v\in\Gamma_a} W_{a,v}^{(t)}Z_{v\to a}^{(t)} \qquad (16)\end{aligned}$$

For the first term:

$$\begin{aligned}\frac{1}{n}\sum_{a=1}^n\sum_{v\in\Gamma_a} W_{a,v}^{(t)}x_v^{(t+1/2)} &= \frac{1}{n}\sum_{a=1}^n W_a^{(t)}x^{(t+1/2)}\\ &= \frac{1}{n}\mathbf{1}^\intercal x^{(t+1/2)}\\ &= \bar{x}^{(t+1/2)}\end{aligned}$$

Where we used the properties of the mixing matrix.

Focusing on the second term in (16), we obtain:

$$\begin{aligned}\frac{1}{n}\sum_{a=1}^n\sum_{v\in\Gamma_a} W_{a,v}^{(t)}Z_{v\to a}^{(t)} &= \frac{1}{n}\sum_{a=1}^n\sum_{v=1}^n W_{a,v}^{(t)}Z_{v\to a}^{(t)}\\ &= \frac{1}{n}\sum_{v=1}^n\sum_{a=1}^n W_{a,v}^{(t)}Z_{v\to a}^{(t)}\\ &= 0.\end{aligned}$$

Plugging this into (16) yields the desired result:

$$\bar{x}^{(t+1)} = \frac{1}{n}\sum_{a=1}^n x_a^{(t+1)} = \bar{x}^{(t+1/2)}$$

□

Lemma 3.3. Noise characterization for Algorithm 1. *Consider that for node a, for all $v \in \Gamma_a^{(t)}$, $Y_{a\to v}^{(t)} \sim \mathcal{N}\left(0, \gamma^2\varsigma_a^2\right)$, for a fixed topology $W^{(t)}$. Then, using the definition of Algorithm 1, we have:*

$$\forall a,v \in [[1,n]], Z_{a\to v}^{(t)} \sim \mathcal{N}\left(0, (\sigma_{a\to v}^{(t)})^2\right)$$

with

$$(\sigma_{a\to v}^{(t)})^2 = \left(\frac{(d_a-1)^2}{d_a^2} + \frac{\sum_{j\in\Gamma_a^{(t)}, j\neq v}(W_{a,j}^{(t)})^2}{(d_a W_{a,v}^{(t)})^2}\right)\gamma^2\varsigma_a^2.$$

Proof. First, looking at the definition of $Z_{a\to v}$, we obtain that:

$$\begin{aligned}Z_{a\to v} &= Y_{a\to v} - \frac{1}{d_a W_{a,v}}\sum_{j\in\Gamma_a} W_{a,j}Y_{a\to j}\\ &= \frac{d_a-1}{d_a}Y_{a\to v} - \frac{1}{d_a W_{a,v}}\sum_{\substack{j\in\Gamma_a\\ j\neq v}} W_{a,j}Y_{a\to j} \qquad (17)\end{aligned}$$

Thus, $Z_{a\to v}$ is a linear combination of independent Gaussian noises. This means that $Z_{a\to v}$ also follows a Gaussian distribution. Since the mean of all $Y_{a\to v}$ is 0, so is the mean of $Z_{a\to v}$.

To obtain the desired result, we only need to look at the variance. Using (17), we obtain:

$$\begin{aligned}\mathbb{V}(Z_{a\to v}) &= \mathbb{V}(\frac{d_a-1}{d_a}Y_{a\to v} - \frac{1}{d_a W_{a,v}}\sum_{\substack{j\in\Gamma_a\\ j\neq v}} W_{a,j}Y_{a\to j})\\ &= (\frac{d_a-1}{d_a})^2\,\mathbb{V}(Y_{a\to v}) + (\frac{1}{d_a W_{a,v}})^2\,\mathbb{V}\left(\sum_{\substack{j\in\Gamma_a\\ j\neq v}} W_{a,j}Y_{a\to j}\right)\\ &= (\frac{d_a-1}{d_a})^2\gamma^2\varsigma_a^2 + (\frac{1}{d_a W_{a,v}})^2\sum_{\substack{j\in\Gamma_a\\ j\neq v}}(W_{a,j})^2\gamma^2\varsigma_a^2\\ &= \left(\frac{(d_a-1)^2}{d_a^2} + \frac{\sum_{\substack{j\in\Gamma_a\\ j\neq v}}(W_{a,j})^2}{(d_a W_{a,v})^2}\right)\gamma^2\varsigma_a^2\end{aligned}$$

□

D Convergence rate of Zip-DL

D.1 Useful inequalities

Lemma D.1. *For any set of n vectors $(a_i)_{i=1}^n$, $a_i \in \mathbb{R}^d$:*

$$\left\|\sum_{i=1}^n a_i\right\|^2 \leq n\sum_{i=1}^n \|a_i\|^2$$

Lemma D.2. *For any vectors $\mathbf{a},\mathbf{b} \in \mathbb{R}^d$, for any $\beta > 0$, we have:*

$$\|\mathbf{a}+\mathbf{b}\|^2 \leq (1+\beta)\|\mathbf{a}\|^2 + (1+\beta^{-1})\|\mathbf{b}\|$$

D.2 Convergence rate results

Theorem 4.5. Convergence rate of Zip-DL. *For any number of iterations T, there exists a constant stepsize γ s.t. for Algorithm 2, $\frac{1}{2W_T}\sum_{t=0}^T w_t(\mathbb{E}\left[f(\bar{x}^{(t)})\right] - f^*) + \frac{\mu}{2}r_{T+1}$ is bounded by:*

$$O\left(\frac{\bar{\omega}^2}{n\mu T} + \frac{LA'}{\mu^2T^2} + \frac{r_0 L}{p}\exp\left[-\frac{\mu p(T+1)}{192\sqrt{3}L}\right]\right),$$

where $A' = \frac{16-4p}{2(16-7p)}(\bar{\omega}^2 + \frac{18}{p}\bar{\vartheta}^2) + \frac{d}{n}\frac{16-4p}{16-7p}\sum_{a,v=1}^n d_a\frac{(d_v-1)^2}{d_v}\varsigma_v^2$, $f^* = f(x^*)$, $r_t = \mathbb{E}\left[\left\|\bar{x}^{(t)} - x^*\right\|^2\right]$, $w_t = (1-\frac{\mu}{2}\gamma)^{-(t+1)}$ and $W_T = \frac{1}{T}\sum_{t=1}^T w_t$.

Proof. (Theorem 4.5) We used a similar situation to [23] with $\tau = 1$ and a fixed communication matrix sampling distribution. The proof follows the same structure as in their paper. Our algorithm only induces some changes in some of the intermediary lemmas that need to be adapted to obtain the main result.

To this end, we restate Proposition D.3 and Lemmas D.4 and 4.7 in our setting. We can then solve the main equation in the following manner:

- We bound the distance of the averaged model to the optimum Lemma D.4. It is the case $r_t = \mathbb{E}\left[\left\|\bar{x}^{(t)} - x^*\right\|^2\right]$, $e_t = f(\bar{x}^{(t)}) - f(x^*), a = \frac{\mu}{2}, b = 1, c = \frac{\bar{\omega}^2}{n}$ and $B = 3L$
- We also bound the consensus distance with a recursive bound using Lemma 4.7. The next step is to determine the precise constants to continue the proof.

The equation of the consensus distance (Lemma 4.7) is of the following form:

$$\Xi_t \leq (1+\beta)(1-\frac{7p}{16})\Xi_{t-1} + (1+\beta)D\gamma^2 e_{t-1} + \left((1+\beta)A + (1+\beta^{-1})\frac{d}{n}\sum_{i=1}^{n} d_i \sum_{v=1}^{n} \frac{(d_v-1)^2}{d_v}\varsigma_v^2\right)\gamma^2$$

with $e_t = f(\bar{x}^{(t)}) - f(x^*)$, $D = \frac{36L}{p}$ and $A = \bar{\omega}^2 + \frac{18}{p}\bar{\vartheta}^2$

Because of the $1+\beta$ factor, we cannot directly apply the recursion-solving Lemma to our scenario (Lemma 12 in [23]). We can however modify our current equation to match the beginning of their proof of this Lemma. This is mostly possible because we are in the case $\tau = 1$, meaning that we require a slightly stronger property on the matrices' distribution.

We can now rewrite the previous equation by setting $\beta = \frac{3p}{16-7p}$ (rq: we only require $\beta > 0$, which is satisfied since $0 \leq p \leq 1$),

$$(1+\beta) = \frac{16-7p+3p}{16-7p} = \frac{16-4p}{16-7p}$$

and

$$(1+\beta)(1-\frac{7p}{16}) = \frac{16-4p}{16-7p}\frac{16-7p}{16} = \frac{16-4p}{16} = 1-\frac{p}{4}$$

Putting these inside the main equation, and setting

$$A' = \left((1+\beta)A + (1+\beta^{-1})\frac{d}{n}\sum_{i=1}^{n} d_i \sum_{v=1}^{n} \frac{(d_v-1)^2}{d_v}\varsigma_v^2\right)\gamma$$
$$D' = \frac{1}{2}(1+\beta)D = \frac{16-4p}{2(16-7p)}\frac{36L}{p}$$

we obtain:

$$\Xi_t \leq (1-\frac{p}{4})\Xi_{t-1} + 2D'\gamma^2 e_{t-1} + 2A'\gamma^2$$

This is exactly the term obtained in [23]'s Lemma 12 after unrolling the different terms, which is only needed when $\tau > 1$. Thus, in our case, we can fall back to their proof using this approach. We just need to ensure Lemma 12's hypothesis are verified:

- $0 < p \leq 1$
- $\tau = 1 \geq 1$
- $A', D' \geq 0$
- $\{\gamma^2\}_{t\leq 0}$ is a $\frac{8}{p}$-slow decreasing sequence since it is a constant.
- $\{w_t := (1-a\gamma)^{-(t+1)}\}$ is a $\frac{16}{p}$-slow increasing sequence of weights.

Thus, we can have the same reasoning as the proof of Lemma 12 in [23], and obtain the lemma's result with the following equation:

$$B\sum_{t=0}^{T} w_t \Xi_t \leq \frac{b}{2}\sum_{t=0}^{T} w_t e_t + 64A'B\gamma^2\sum_{t=0}^{T} w_t \tag{18}$$

for some constant E and stepsize $\gamma \leq \frac{1}{16}\sqrt{\frac{pb}{D'B}}$

From this point on, we can follow the exact ending of the proof, the only difference are our new constants A' and D'. We thus obtain:

$$\frac{1}{2W_T}\sum_{t=0}^{T} b w_t e_t \leq \frac{1}{W_T}\sum_{t=0}^{T}\left(\frac{(1-a\gamma)w_t}{\gamma}r_t - \frac{w_t}{\gamma}r_{t+1}\right) + \frac{c}{W_T}\sum_{t=0}^{T} w_t\gamma + \frac{64BA'}{W_T}\sum_{t=0}^{T} w_t\gamma^2$$

(with $W_T = \sum_{t=0}^{T} w_t$).

Finally, we use Lemma 13 of [23] to obtain the final result, since we verify the following hypothesis: $a, b > 0, c, A', B \geq 0$

Thus, we obtain that for a well-chosen γ:

$$\frac{1}{2W_T}\sum_{t=0}^{T} b e_t w_t + a r_{T+1} \leq O\left(r_0 d \exp\left[-\frac{a(T+1)}{d}\right] + \frac{c}{aT} + \frac{BA'}{a^2T^2}\right).$$

Plugging in the values yields the result for Theorem 4.5.

□

From the previous result, we also prove the convergence rate to an arbitrary ρ accuracy:

Corollary 4.6. *Setting all the constants to be the same as in Theorem 4.5, for any target accuracy $\rho > 0$, there exists a constant stepsize γ such that Algorithm 2 reaches the target accuracy after at most*

$$\frac{3\kappa\bar{\omega}^2}{n\mu\rho} + \sqrt{\frac{3\kappa LA'}{\rho\mu^2}} + \frac{192\sqrt{3}L}{\mu p}\ln\left[\frac{3\kappa r_0 L}{\rho p}\right]$$

training iterations, where κ is the constant that arises when upper bound $O\left(\frac{\bar{\omega}^2}{n\mu T} + \frac{LA'}{\mu^2T^2} + \frac{r_0 L}{p}\exp\left[-\frac{\mu p(T+1)}{192\sqrt{3}L}\right]\right)$ is expanded out.

Proof. For Algorithm 2 to reach the target accuracy ρ, we need to have:

$$\frac{1}{2W_T}\sum_{t=0}^{T} w_t\left(\mathbb{E}\left[f(\bar{x}^{(t)})\right] - f^*\right) + \frac{\mu}{2}r_{T+1} \leq \rho \tag{19}$$

However, from Theorem 4.5, we know that

$$\frac{1}{2W_T}\sum_{t=0}^{T} w_t\left(\mathbb{E}\left[f(\bar{x}^{(t)})\right] - f^*\right) + \frac{\mu}{2}r_{T+1} \leq \kappa\left(\frac{r_0 L}{p}\exp\left[-\frac{\mu p(T+1)}{192\sqrt{3}L}\right] + \frac{\bar{\omega}^2}{n\mu T} + \frac{LA'}{\mu^2T^2}\right)$$

for some constant $\kappa > 0$.

Thus, in order to satisfy (19), it suffices to simultaneously have:

$$\kappa\frac{r_0 L}{p}\exp\left[-\frac{\mu p(T+1)}{192\sqrt{3}L}\right] \leq \frac{\rho}{3}$$
$$\iff \exp\left[\frac{\mu p(T+1)}{192\sqrt{3}L}\right] \geq \frac{3\kappa r_0 L}{\rho p}$$
$$\iff T \geq \frac{192\sqrt{3}L}{\mu p}\ln\left[\frac{3\kappa r_0 L}{\rho p}\right] - 1, \tag{20}$$

$$\kappa\frac{\bar{\omega}^2}{n\mu T} \leq \frac{\rho}{3} \iff T \geq \frac{3\kappa\bar{\omega}^2}{n\mu\rho}, \tag{21}$$

and

$$\kappa\frac{LA'}{\mu^2T^2} \leq \frac{\rho}{3} \iff T \geq \sqrt{\frac{3\kappa LA'}{\rho\mu^2}}. \tag{22}$$

Therefore, in order to simultaneously satisfy the inequalities in (20),(21), and (22), it suffices to have

$$T \geq \frac{192\sqrt{3}L}{\mu p}\ln\left[\frac{3\kappa r_0 L}{\rho p}\right] - 1 + \frac{3\kappa\bar{\omega}^2}{n\mu\rho} + \sqrt{\frac{\kappa LA'}{3\mu^2}}$$
$$\implies T > \frac{192\sqrt{3}L}{\mu p}\ln\left[\frac{3\kappa r_0 L}{\rho p}\right] + \frac{3\kappa\bar{\omega}^2}{n\mu\rho} + \sqrt{\frac{3\kappa LA'}{\rho\mu^2}} \qquad \square$$

D.3 Intermediary lemmas proofs

We state and prove the necessary lemmas for the convergence proof of Section D.2.

Proposition D.3. Mini-batch variance *(Proposition 5 in [23]) Assume that F_i is L-smooth (Assumption 4.1) with bounded noise at the optimum (Assumption 4.3). Then, for any $i \in [\![1, n]\!]$, we have:*

$$\mathbb{E}_{\xi_1,\dots,\xi_n}\left[\left\|\frac{1}{n}\sum_{i=1}^{n}(\nabla f(x_i) - \nabla F_i(x_i,\xi_i))\right\|^2\right]$$
$$\leq \frac{3L^2}{n}\sum_{i=1}^{n}\left\|x_i - \bar{x}\right\|^2 + 6L(f(\bar{x}) - f(x^*)) + 3\bar{\omega}^2.$$

Proof. (Proposition D.3) Nothing changes in this proof compared to the original work, since only the gradient and the loss functions are needed, and averaging rounds are not considered. $\square$

Lemma D.4. Descent lemma for convex cases. *(Lemma 8 of [23]) Under Assumptions 4.1 to 4.4, with stepsize $\gamma \leq \frac{1}{12L}$ we have:*

$$\mathbb{E}_{\xi_1^{(t)},\dots,\xi_n^{(t)}}\left[\left\|\bar{x}^{(t+1)} - x^*\right\|^2\right] \leq (1 - \frac{\gamma\mu}{2})\left\|\bar{x}^{(t)} - x^*\right\|^2 + \frac{\gamma^2\bar{\omega}^2}{n} - \gamma(f(\bar{x}^{(t)}) - f(x^*)) + \gamma\frac{3L}{n}\sum_{i=1}^{n}\left\|\bar{x}^{(t)} - x_i^{(t)}\right\|^2.$$

Proof. (Lemma D.4) Because of Zip-DL's properties (in particular Corollary 3.2), this property holds almost immediately from Lemma 8 of [23]. Using Corollary 3.2, we have:

$$\left\|\bar{x}^{(t+1)} - x^*\right\|^2 = \left\|\bar{x}^{(t+1/2)} - x^*\right\|^2 = \left\|\bar{x}^{(t)} - \frac{\gamma}{n}\sum_{i=1}^{n}\nabla F_i(x_i^{(t)},\xi_i^{(t)}) - x^*\right\|^2$$

This corresponds to the first line of Lemma 8, so following the proof will yield the same result. More generally, this property would not hold as it stands for a method that only cancels the noise in expectation: because we consider a norm here, this will lead to an additional term equal to the variance of the residual noise on the network, e.g. the variance of the sum of all the noises. If the noises are not correlated, this is an estimator of the original distribution, yielding an additional term. In our case, this term is exactly zero. $\square$

Lemma 4.7. (Recursion for consensus distance) *Under Assumptions 4.1 to 4.4, if stepsizes $\gamma \leq \frac{p}{96\sqrt{3}L}$, then for any $\beta > 0$:*

$$\Xi_t \leq (1+\beta)\left(1 - \frac{7p}{16}\right)\Xi_{t-1} + \gamma^2(1+\beta)\left(\bar{\omega}^2 + \frac{18}{p}\bar{\vartheta}^2\right) + (1+\beta)\frac{36L}{p}\left(f(\bar{x}^{(t-1)}) - f(x^*)\right) + \gamma^2(1+\beta^{-1})\frac{d}{n}\sum_{i=1}^{n}d_i\sum_{v=1}^{n}\left(\frac{(d_v-1)^2}{d_v}\varsigma_v^2\right),$$

where $\Xi_t = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_t\left[\left\|x_i^{(t)} - \bar{x}^{(t)}\right\|^2\right]$ is the consensus distance

This lemma has an additional last term compared to state-of-the-art DL [23]. It stems from the presence of noise, that shifts local models away from the true average.

Proof. (Lemma 4.7)

$$n\Xi_t = \sum_{i=1}^{n}\mathbb{E}_t\left[\left\|x_i^{(t)} - \bar{x}^{(t)}\right\|^2\right] = \sum_{i=1}^{n}\mathbb{E}_t\left[\left\|(x_i^{(t)} - \bar{x}^{(t-1)}) - (\bar{x}^{(t)} - \bar{x}^{(t-1)})\right\|^2\right] \leq \sum_{i=1}^{n}\mathbb{E}_t\left[\left\|(x_i^{(t)} - \bar{x}^{(t-1)})\right\|^2\right]$$

Where we used that $\sum_{i=1}^{n}\|a_i - \bar{a}\|^2 \leq \sum_{i=1}^{n}\|a_i\|^2$. Unrolling the model update:

$$\begin{aligned} x_i^{(t)} &= \sum_{v\in\Gamma_i^{(t-1)}} W_{i,v}^{(t-1)}(x_v^{(t-1/2)} + Z_{v\to i}^{(t-1)}) \\ &= \sum_{v\in\Gamma_i^{(t-1)}} W_{i,v}^{(t-1)}((x_v^{(t-1)} - \gamma\nabla F_v(x_v^{(t-1)},\xi_v^{(t-1)})) + Z_{v\to i}^{(t-1)}) \\ &= \sum_{v\in\Gamma_i^{(t-1)}}(W_{i,v}^{(t-1)}(x_v^{(t-1)})) + \sum_{v\in\Gamma_i^{(t-1)}}(W_{i,v}^{(t-1)}Z_{v\to i}^{(t-1)}) \\ &\quad - \sum_{v\in\Gamma_i^{(t-1)}}(W_{i,v}^{(t-1)}\gamma\nabla F_v(x_v^{(t-1)},\xi_v^{(t-1)})) \end{aligned}$$

This yields, after expanding the recursion and using Lemma D.2, for any $\beta > 0$:

$$n\Xi_t \leq (1+\beta)\underbrace{\sum_{i=1}^{n}\mathbb{E}_t\left[\left\|\sum_{v\in\Gamma_i^{(t-1)}} T_3\right\|^2\right]}_{:=T_1} + (1+\beta^{-1})\underbrace{\sum_{i=1}^{n}\mathbb{E}_t\left[\left\|\sum_{v\in\Gamma_i^{(t-1)}} (W_{i,v}^{(t-1)} Z_{v\to i}^{(t-1)})\right\|^2\right]}_{:=T_2}$$

where we have

$$T_3 := W_{i,v}^{(t-1)}\left(x_v^{(t-1)} - \gamma\nabla F_v(x_v^{(t-1)}, \xi_v^{(t-1)})\right) - \bar{x}^{(t-1)}$$

Looking at the second term, and using Lemma D.1:

$$\begin{aligned}
T_2 &\leq \sum_{i=1}^{n} d_i \sum_{v\in\Gamma_i^{(t-1)}} \mathbb{E}_t\left[\left\|W_{i,v}^{(t-1)} Z_{v\to i}^{(t-1)}\right\|^2\right] \\
&\leq \sum_{i=1}^{n} d_i \sum_{v\in\Gamma_i^{(t-1)}} \mathbb{E}_t\left[(W_{i,v}^{(t-1)})^2\left\|Z_{v\to i}^{(t-1)}\right\|^2\right] \\
&\leq \sum_{i=1}^{n} d_i \sum_{v\in\Gamma_i^{(t-1)}} \mathbb{E}_{t,i\in\Gamma_v^{(t-1)}}\left[(W_{i,v}^{(t-1)})^2\left\|Z_{v\to i}^{(t-1)}\right\|^2\right] \\
&\leq \sum_{i=1}^{n} d_i \sum_{v\in\Gamma_i^{(t-1)}} \mathbb{E}_{t,i\in\Gamma_v^{(t-1)}}\left[(W_{i,v}^{(t-1)})^2\mathbb{E}_{W^{(t-1)}}\left[\left\|Z_{v\to i}^{(t-1)}\right\|^2\right]\right]
\end{aligned}$$

Using Lemma 3.3 for a fixed gossip matrix, and leveraging $W_{i,v}^{(t)} = W_{v,i}^{(t)}$ since we assume symmetric matrices, we obtain:

$$\begin{aligned}
T_2 &\leq \sum_{i=1}^{n} d_i \sum_{v=1}^{n} \mathbb{E}_{t,i\in\Gamma_v^{(t-1)}}\left[(W_{v,i}^{(t-1)})^2 d(\sigma_{v\to i}^{(t-1)})^2\right] \\
&\leq d\gamma^2 \sum_{i=1}^{n} d_i \sum_{v=1}^{n}\left(\frac{(d_v-1)^2}{d_v^2} + \frac{d_v-1}{d_v^2}\right)\varsigma_v^2 \\
&\leq d\gamma^2 \sum_{i=1}^{n} d_i \sum_{v=1}^{n}\left(\frac{(d_v-1)^2}{d_v}\right)\varsigma_v^2
\end{aligned}$$

Where we used that $(W_{i,v})^2 \leq 1$ for all $i, v \in \mathcal{V}$.

For T_1, we obtain that:

$$T_1 = \mathbb{E}_t\left[\left\|W^{(t-1)}\left(x^{(t-1)} - \gamma\nabla F(x^{(t-1)}, \xi^{(t-1)})\right) - \bar{x}^{(t-1)}\right\|_F^2\right]$$

This is the exact notation from [23], in the proof of the corresponding Lemma (Lemma 9), with the notation $\tau = 1$ (our matrix notation are transposed to theirs). By following the same steps, we obtain:

$$\begin{aligned}
T_1 \leq& n(1-\frac{p}{2})\Xi_{t-1} + n\frac{p}{16}\Xi_{t-1} + n(\bar{\omega}^2 + \frac{18}{p}\bar{\vartheta}^2)\gamma^2 \\
&+ n\frac{36L}{p}\gamma^2(f(\bar{x}^{(t-1)}) - f(x^*))
\end{aligned}$$

Plugging T_1 and T_2 back into the original term, we obtain:

$$\begin{aligned}
\Xi_t \leq& (1+\beta)\left((1-\frac{7p}{16})\Xi_{t-1} + \frac{36L}{p}\gamma^2(f(\bar{x}^{(t-1)}) - f(x^*))\right) \\
&+ (1+\beta)\left((\bar{\omega}^2 + \frac{18}{p}\bar{\vartheta}^2)\gamma^2\right) \\
&+ (1+\beta^{-1})\frac{d\gamma^2}{n}\sum_{i=1}^{n} d_i \sum_{v=1}^{n}\left(\frac{(d_v-1)^2}{d_v}\right)\varsigma_v^2 \\
\leq& (1+\beta)(1-\frac{7p}{16})\Xi_{t-1} + (1+\beta)\frac{36L}{p}\gamma^2(f(\bar{x}^{(t-1)}) - f(x^*)) \\
&+ \gamma^2(1+\beta)(\bar{\omega}^2 + \frac{18}{p}\bar{\vartheta}^2) \\
&+ \gamma^2\left((1+\beta^{-1})\frac{d}{n}\sum_{i=1}^{n} d_i \sum_{v=1}^{n}\left(\frac{(d_v-1)^2}{d_v}\varsigma_v^2\right)\right)
\end{aligned}$$

For any $\beta > 0$, which is the desired result. □