# Differential Privacy for Decentralized Learning

Edwige Cyffers

Université de Lille
Ecole Doctorale MADIS

# Thèse de Doctorat

Spécialité **Informatique**

présentée par
**Edwige Cyffers**

---

# Differential Privacy for Decentralized Learning

---

# Confidentialité différentielle pour l'apprentissage décentralisé

---

Soutenue publiquement à **Paris**, le **05/12/2024** devant le jury composé de

| | | | |
|---|---|---|---|
| M^me Virginia **Smith** | Professeure | CMU | Rapportrice |
| M Antti **Honkela** | Professeur | University of Helsinki | Rapporteur |
| M^me Rachel **Cummings** | Professeure | Columbia University | Examinatrice |
| M Hadrien **Hendrikx** | Chargé de Recherche | Inria | Examinateur |
| M. Pierre **Senellart** | Professeur | ENS | Président du jury |
| M. Aurélien **Bellet** | Directeur de Recherche | Inria | Directeur de thèse |
| M. Peter **Kairouz** | Chercheur | Google | Invité |

---

Centre de Recherche en Informatique, Signal et Automatique de Lille (CRIStAL),
UMR 9189 Équipe Magnet, 59650, Villeneuve d'Ascq, France





# Acknowledgements

Acknowledgements stand out as the most private yet most read part of a thesis. I hoped for a well-written, carefully proofread text, but this is being written a few minutes before printing. To the future graduate: stop procrastinating by reading this; write yours. To anyone I fail to name below, thank you, and thank you again for your forgiveness.

**To Aurélien** Not only did you give me your trust, a lot of freedom, and incredible feedback for all the work, including this thesis, but I was also a witness to your personal growth—from your HDR at the beginning of my thesis to your promotion to DR, moving to Montpellier, becoming a father, participating in many big projects. I believe I learned a lot from your example. You have been the supervisor I was looking for.

**To the jury** Thank you, Virginia, for agreeing to write a report for my thesis; it is a great honor. Thank you, Antti, for also accepting and for the very precise review you provided. When we first met randomly at ICLR and you recognized me as one of the people who reviewed papers under your area chair guidance, I knew within minutes I hoped you would review my thesis. Thank you, Rachel, for agreeing to join my jury: as I was inspired by your talk at the beginning of my PhD, I am truly happy you accepted. Thank you, Hadrien: you worked with nearly all my co-authors, and I hope the work presented here interested you. Thank you Pierre for accepting to chair the jury. Finally, thank you, Peter, for completing this jury despite the early schedule it required of you.

**To my coauthors** Thank you, Mathieu, for your wonderful intuition and climbing work sessions. Deb, for taming or warning against nasty computations with complex handwaving before writing on the whiteboard. Jalaj, for not only scientific discussions but also those on Maupassant, my future in research, and French food—I hope you come live in Europe. Paul, for your chill; Constantin, for your humble curiosity; and Aymeric, for your energy during Flamby. Abdellah, for being patient as my first supervision attempt. Jamal Atif, for the discussions; Olivier Cappé, for all the meetings at the whiteboard—I hope to one day master this skill and have learned much from you.

**To Magnet and others** Thanking Marc extensively is a topos in all theses from this team. Yet, I must follow tradition, as your role in Magnet cannot be overstated. Thank you for always being there when it matters, solving problems, and managing the team humanely. Thanks also to small Marc for the discussions, Batiste for support, and Mikaela for glimpses into NLP. To Aurore, thank you for sorting out logistics and administrative issues. To Leo, for mentorship. I also thank all my students over the years: if you had half the fun I had and learn half of what you taught me, I consider my job done. To NeurIPS in Paris organizers, in particular Linus and Pierre.

**To people met during my education** Another topos is thanking teachers. I cannot do so as I prefer honesty: while many friends had strong mentoring relationships, I did not experience this. Likely due to my behavior, I struggled to worship teachers or please them, but I also suspect gender stereotypes played a role. I was more praised for litterature, or even for my clothes. I wish I could thank those who protected me from harassment or aggression, but those responsible fell short. To those who ask why there are few women in science — especially those suggesting biological reasons—thank you for asking. I was fortunate to feel special thanks to EGMO, granting me access to "prépa." I am grateful to the L'Oréal-UNESCO prize, which allowed me to meet amazing awardees. I am the last to defend my thesis from the 2023 cohort, a group far more experienced than I, which gives me optimism. Exceptional people crossed my path: Azadeh, like a sister; Nell, who inspired me to improve; Wenqi, for unconditional support and leisure; Claire, for seriousness and arts; Lucas, for teaching me how to work; Lucas V, for shared colles; Eva, for kindness and analysis. And the rest of "les filles" and LLG people.

**To the normaliens** For all the birthday parties. Special mention to the best Maxime for all our projects and adventures, to Antonin for the peculiar and beautiful view on this world, and to Loïc for loyal and constant friendship.

**To women** I would like to thank the women who came before me, who fought for our rights and made possible and mundane the fact of being in control of my body, having access to the same rights and positions as men, including in research. Marie Curie is often cited as a role model, but the fact that she could win two Nobel Prizes, save lives during war, raise a child who also won a Nobel Prize, and yet be left aside by the Nobel Committee until her husband demanded her inclusion for the first Prize, or be asked not to attend the second ceremony because she had a relationship with a married man, feels more like a deterrent than an encouragement. I prefer to thank the writers who provided me with strength over the years: Charlotte Brontë, Virginia Woolf, Annie Ernaux, Virginie Despentes, Nathalie Azoulai, Sally Rooney, Ursula K. Le Guin, Hannah Arendt, and Simone de Beauvoir. Please consider reading them. Closer to me, I would like to thank the few senior women who crossed my path. I hope

future female PhD students will have similar or even more guidance: Clarisse Dhaenens for administrative support, Mikaela again, Anne Canteaut for the few discussions, Julia Kempe for the months at CSD, and Celestine Mendler-Dünner for discussions on performative learning and ethical implications of ML. I especially highlight Claire Vernade, who over the years acted as a mentor, giving me a lot of precise and important advice during my thesis. I truly value this sorority.

**To institutions and practicalities** This work was supported by the ANR-20-THIA-0014 program "AI PhD@Lille" and Région Hauts-de-France. Thank you to the Linux ecosystem, the Internet, Wikipedia, databases allowing free access to articles, Thunderbird, Mathpix, Remarkable, ENS for funding my studies and sparing me student debt, and TGVmax for avoiding planes and reducing costs. I thank Institut Henri Poincaré for hosting my defense.

**To Horses** Horses taught me that training complex neural networks is challenging and fascinating. The medals, like papers, are noisy, sparse rewards hiding the long process of understanding. What matters is the path, and one who enjoys searching for answers can appreciate both horse riding and research. Holdup proved more challenging than expected, but I hope we succeed before tenure! Riding also brought people like Agnès Décrion, the first adult mentor. For the writing period, thanks to Thaïs and Stéphane Langlois, proving dreams can defeat death.

**To my family** Thank you for your pride, logistic support, and love. To Alain: love is the best proof that needing privacy is not shameful but a testament to building something so precious it demands intimacy.

# Résumé

L'effondrement des coûts de stockage et de traitement des données, conjugué à l'essor de la numérisation, a permis de nouvelles applications et possibilités pour l'apprentissage automatique. En pratique, les Big Data vont souvent de pair avec la collecte de données sensibles. Ainsi, la protection de la vie privée, notamment la prévention des fuites de données intentionnelles ou accidentelles, est l'un des principaux défis de l'intelligence artificielle digne de confiance. Une première approche pour une meilleure maitrise des données consiste à les conserver de manière décentralisée, en ne partageant que les informations nécessaires pour le processus d'apprentissage. Cela peut être réalisé soit via un serveur central orchestrant le processus dans l'apprentissage fédéré, soit à travers des communications pair-à-pair. Cependant, cela ne garantit pas que les données sont protégées tout au long du processus, l'apprentissage fédéré étant connu pour être vulnérable aux attaques de reconstruction, qui permettent de reconstruire partiellement ou totalement les données en exploitant le modèle, sans avoir directement accès aux données locales elles-mêmes. Pour quantifier et contrôler de manière fiable la perte de confidentialité, la confidentialité différentielle est actuellement la référence dans la recherche et l'industrie pour les applications d'apprentissage automatique.

Dans cette thèse, nous nous situons à l'intersection entre l'apprentissage automatique, les algorithmes décentralisés et la confidentialité différentielle. Nous introduisons la première attaque de reconstruction en apprentissage décentralisé, prouvant la capacité d'exploiter les fuites de confidentialité entre participants non directement connectés entre eux, ce qui prouve la nécessité d'inclure des mécanismes de défense dans l'apprentissage décentralisé. Nous introduisons ensuite une nouvelle variante de la confidentialité différentielle, la Network Differential Privacy, adaptée à l'apprentissage décentralisé où chaque nœud ne voit que les communications locales. À l'aide de cette variante, nous analysons les garanties de confidentialité et d'utilité de divers algorithmes décentralisés, notamment les algorithmes de gossip et les marches aléatoires pour la descente de gradient stochastique et l'ADMM. Nos contributions démontrent que la décentralisation peut amplifier la confidentialité dans le cadre de la confidentialité différentielle, et que les gains dépendent de l'algorithme et du graphe de communication. Cela ouvre la voie à l'utilisation de la décentralisation comme outil pour développer des méthodes d'apprentissage automatique protégeant mieux la vie privée.

# Abstract

The collapse of storage and data processing costs, along with the rise of digitization, has brought new applications and possibilities to machine learning. In practice, Big data is often synonymous with sensitive data collection. Hence, protecting privacy— especially by avoiding data leakage, intentional or accidental —is one of the key challenges in Trustworthy Machine Learning. A first direction towards more control over data is to keep it decentralized, exchanging only the information needed to run the learning process. This can be achieved through a central server orchestrating the learning process in federated learning or through peer-to-peer communications. However, this does not guarantee that data is protected throughout the entire process, as federated learning is known to be vulnerable to privacy attacks. To reliably quantify and control the privacy loss occurring in machine learning algorithms, Differential Privacy is currently the gold standard both in research and industry for machine learning applications.

This thesis lies at the intersection of machine learning, decentralized algorithms and differential privacy. We present the first reconstruction attack in decentralized learning, targeting privacy leaks among participants not directly connected, proving the need to include defense mechanisms in this setting. We then introduce a new variant of differential privacy, Network Differential Privacy, which is suited for decentralized learning where each node only sees local communications. Using this variant, we analyze the privacy and utility guarantees of various decentralized algorithms, namely gossip algorithms and random walks for stochastic gradient descent, and ADMM. Our contributions demonstrate that decentralization can bring privacy amplification in the sense of differential privacy, and that the gains depend on the algorithm and the communication graph. This paves the way for the use of decentralization as a tool to develop more effective privacy-preserving machine learning.

# Table of Contents

# Chapter 1

# Introduction

## 1.1 Context

A short time before my birth, my parents joined a French panel of consumers and participated for years. It involved transmitting all the information about the goods we bought: when returning home from the stores, my father sat with a bar code scanner, going through all the receipts, and I had to find each item for scanning before storing them in the cupboard or the fridge. Once used to the process, with good organization the overhead was minimal. The interface improved over the years. At the beginning, when a wrong price was entered, someone would call and ask if it was a mistake and if we could retrieve the correct value: this was how outliers and missing data were dealt with. I was told that from all the numbers we entered in the bar code reader, some smart folks were able to predict the future state of the stores. This, of course, was something I couldn't assess, and it sounded quite magical. As I grew up, I was less prone to help with this intrusive process, and even found some extra tasks such as sending out precise weight and body measurements a bit embarrassing. However, this program also had advantages: each data collected earned points, that were converted into attractive goods. For example, we received three bikes through the program, which significantly enhanced our vacation. The benefits were thus significant for us: we would not have given our data away for free, and only the explicit conversion into tangible goods convinced my parents to accept the burden of data collection.[1]

This experience embodies the pre-Big Data era: data collection was painful, costly, and prone to errors. Data curation was a manual process and could incur delays. Precision was limited, only tabular data was used, and, obviously, the designers of the vegetables table believed that all apples were the same, ignoring the various species. The amount of data was limited to a few

[1] I am not advocating for a data for sale model, but describing a scheme I have been exposed to.

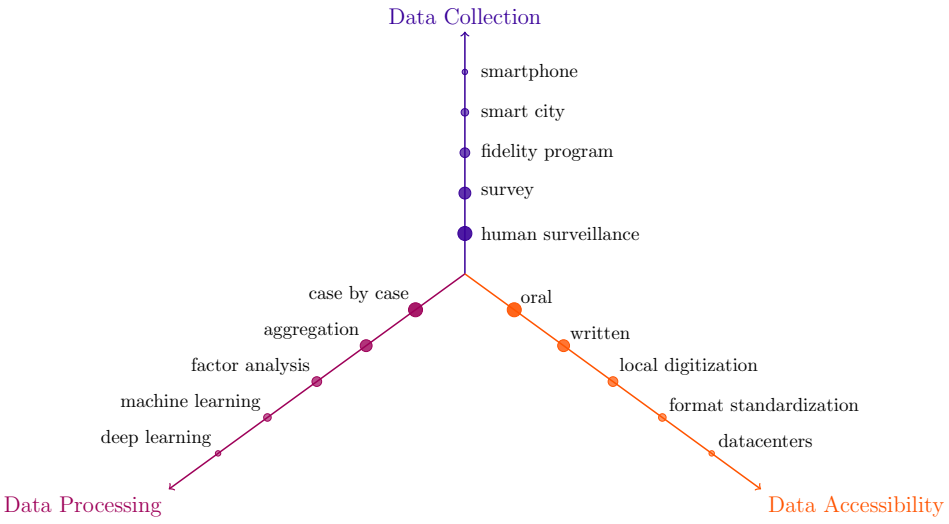



**Figure 1.1** – 3D representation of the changes happening with Big Data.

thousand families, and the compensation for each bit of data given was enough to motivate participants. Since then, data has undergone a three-dimensional upheaval (Figure 1.1):

- **Invisibility of Data Collection.** Data collection is now the default choice. Purchased goods are stored in loyalty programs, GPS positions are collected by multiple apps, and video surveillance is ubiquitous. The revolt I had against sharing my body measurements cannot arise so easily anymore because we never see that data being shared with other parties. We do not see a policeman following each of our steps, even though the police can at any time find the location of our smartphone and thus ours. We collect data that was not accessible before, and the number of people who have their data collected continues to grow through the wide adoption of smartphones and Internet of Things.

- **Availability of Data Storage.** We will see that cheap newspapers were already seen in the past as a revolution in data accessibility. Nowadays, databases can be extremely large and complex and can still be easily accessed without a real understanding of data management, with unprecedented bandwidth and low latency.

- **Advances in Data Processing.** Machine learning breaks the limits of correlations that could be humanly discernible, enabling the use of complex relationships between variables that could not be found manually as long as we can describe a good objective function. Computational power has grown in a nearly exponential manner, following Moore's Law and software has also improved significantly. The size of Machine Learning models continues to grow in number of parameters, and demands more complex architecture and tricks to be trained.

Many technologies made possible by this new ecosystem should arguably not be used, such as social credit, massive facial recognition, and targeted advertisement, which only aim to classify people to the benefit of a happy few and create divide and fear of diversity in society [Par12; NC15; Zub19]. However, it can enable a better understanding of our world, and responsible uses are more likely to happen if the agents are not dispossessed of their data but actively participate in a collaborative process while maintaining privacy over their data. This is, in fact, much harder to achieve than intuition would predict, as data can be leaked in many indirect ways, and effective collaboration between distinct entities is complex. It is however crucial if we want to empower people through their data. Holding the data rather than sending it to a third party favors the possibility to switch between service providers, reduces the tendency toward monopoly and consolidation of the first actors to come into a given market, and gives control back to the users [Shu20; Sch16]. It should also come with the right tools to be informed of the risk of indirect privacy leakage: holding data is not a goal *per se*, but a mean to regain control over it. Estimating the various privacy risks is essential for a coherent data policy.

## 1.2 Scope of the thesis

In this thesis, we explore the impact of keeping data decentralized on the privacy properties of machine learning. Given a machine learning task and decentralized datasets held by separate entities, how should we define privacy and what makes an algorithm more privacy-preserving in this context? How should we train a model to be as private and as efficient as the centralized alternative? Can decentralization have synergies with the objective of privacy? Or is it only an obfuscation strategy that does not resist careful auditing? To study these questions, we rely on two main research areas: decentralized learning and differential privacy.

**Decentralized Learning** In a fully decentralized environment, several participants — who may be referred to as nodes, users, or devices depending on the context — each hold a separate database locally. As opposed to federated learning, where communications are centralized by a global server which orchestrates the learning process [McM+17; Kai+21; Li+20], this thesis focuses on decentralized learning, where a graph of communication indicates which nodes can exchange messages [Boy+06; Hen22; Tan+18; Kol+20]. This setting is particularly applicable to the Fediverse [EM22], where social networks are organized by instances, interacting with others such as in Mastodon [Zig+18], Peertube, or Matrix servers. The communication graph is often determined by existing links between entities and may reflect some extent of trust relationships. For instance, these communication schemes can also correspond to partnerships established between different hospitals or to the geometric graphs resulting from Bluetooth connections between smartphones. However, maintaining data decentralization increases the complexity

of learning algorithms by introducing heterogeneity and requiring extra care to ensure global convergence despite local communications without a central orchestrator [Neg+20; Kol+20].

**Differential Privacy** Privacy protection relies on an accurate estimation of privacy risks. This evaluation can be done empirically via the design of attacks or by establishing theoretical privacy guarantees. Attack successes depend on the hypotheses made on the attacker and only provide evidence of existing vulnerabilities but fixes of a given attack may not protect against other attacks in the future. In contrast, Differential privacy [Dwo+06a] quantifies in the worst case how much information about a single entity in the dataset can be leaked through its influence on the outputs of the algorithm, and thus provide theoretical guarantees against all attacks. More precisely, an algorithm $\mathcal{A}$ is differentially private if, for all pairs of datasets $\mathcal{D} \sim \mathcal{D}'$ and every measurable subset $\mathcal{S} \subset Z$, the following inequality holds:

$$\mathbb{P}(\mathcal{A}(\mathcal{D}) \in \mathcal{S}) \leq \exp(\varepsilon)\mathbb{P}(\mathcal{A}(\mathcal{D}') \in \mathcal{S}) + \delta \quad . \tag{1.1}$$

where $\varepsilon$ is the privacy budget and $\delta$ a small quantity allowing some flexibility. Training machine learning models with differential privacy guarantees is challenging due to the numerous interactions required with the data, the complexity of the models, and the high-dimensionality of the data. Privacy-preserving machine learning studies algorithms such as differentially private stochastic gradient descent [Aba+16; BST14; RA12; SCS13] and explores settings that help limit the risk of data leakage during training.

While decentralization and privacy both provide ways to empower people with their data, combining effectively decentralization and privacy is challenging. Prior to this thesis, the dominant approach was reduced to a variant of differential privacy called local differential privacy, which focuses on a worst-case scenario where no one is trusted, impeding the possibility to achieve competitive accuracy regimes in interesting settings. The practicality of privacy-preserving decentralized learning was thus limited by the used definitions, and in particular could not distinguish between the various possible graphs of communication. However, a natural intuition is that decentralization could bring better privacy by removing the omniscient knowledge of central entities and adding randomness. This opposition between intuition and known theoretical results raised several questions.

Is decentralized learning private by nature? Can we adapt differential privacy to decentralized learning? Are some decentralized algorithms more private than others? What is the impact of the communication graph?

## 1.3 Contributions

This thesis answers to the above questions and allows to better understand the synergies between privacy and decentralization. Our key contributions are as follows.

**First attack in decentralized learning** Enforcing privacy is motivated by the risk of privacy leakage due to the training of the model. While it was already known that federated learning presents indirect leakage through updates sent to the server, we propose the first reconstruction attack by a set of participants targeted at other participants who are not necessarily direct neighbors in the communication graph. This work shows that one cannot rely on decentralization alone to protect sensitive data. Therefore, to provide robust privacy guarantees, decentralized algorithms must be combined with additional defense mechanisms such as those based on differential privacy.

**New variant of differential privacy tailored to decentralized algorithms** We introduce Network Differential Privacy and Pairwise Network Differential Privacy, two relaxations of the classical centralized differential privacy that modifies Equation (1.1) to capture the idea that each participant in the graph only observes their local communication. The pairwise version allows to quantify the privacy leakage from any two pairs of nodes depending on their position in the communication graph. These definitions prove to be useful for analyzing various algorithms.

**Study of privacy guarantees of several decentralized algorithms** Using these new differential privacy definitions, we establish improved privacy and utility guarantees for several popular decentralized algorithms. We study gossip algorithms, random walk-based algorithms, and Alternative Direction Method of Multipliers (ADMM). These results are obtained by leveraging new proof techniques, involving small modifications of the algorithms, new privacy amplification results, links with graph theory, and the extension of existing optimization results.

All the papers presented in this thesis have been presented in 5 papers published at AISTATS 2022, NeurIPS 2022, ICML 2023 and ICML 2024. They have their code freely available on GitHub to allow reproducibility and help researchers use it as baselines, and several repositories have indeed been used by follow-up works by other authors.

## 1.4 Outline of the Thesis

The manuscript is organized as follows.

- In Chapter 2, we provide some background on decentralized machine learning. After introducing supervised learning and the empirical risk minimization problem, we give useful mathematical properties for optimization, such as convexity and smoothness, and some convergence results for first-order optimization algorithms. We then move to the decentralized setting by introducing Federated Learning and fully decentralized learning, with gossip and random walk based algorithms.

- In Chapter 3, we motivate the need for privacy-preserving machine learning algorithms before focusing on differential privacy. We discuss its definition and some important properties. Then, we provide some background on differentially private machine learning, and in particular on differentially private stochastic gradient descent and its guarantees.

- In Chapter 4, mainly based on [MCB24], we describe our attack specific to fully decentralized learning. We show that private data locally held by users can be successfully reconstructed by distant nodes, thus motivating the need to incorporate privacy-preserving mechanisms into decentralized algorithms.

- In Chapter 5, we discuss the adaptation of differential privacy to the decentralized setting. We propose Network Differential Privacy and an extension of it, Pairwise Network Differential Privacy. These definitions provide a framework to prove relevant privacy guarantees when studying decentralized algorithms. This chapter is based on [CB22; Cyf+22].

- In Chapter 6, we study gossip algorithms and in particular a variant of Decentralized Stochastic Gradient Descent (D-SGD) under Pairwise Network Differential Privacy and derive privacy and utility results for this setting, showcasing that decentralization can bring an amplification of the privacy given by local addition of noise, based on the results of [Cyf+22].

- In Chapter 7, we study random walk based decentralized gradient descent algorithms, mainly based on [CBU24] and compare this algorithm to the previous approach based on gossip.

- In Chapter 8, we turn towards Alternative Direction Method of Multipliers, by leveraging the general framework of fixed-point algorithms. We also derive differential privacy guarantees, in centralized, federated, and decentralized settings, based on [CBB23].

- In Chapter 9, we conclude and offer perspectives for future work.

## 1.5 Other Contributions

During the duration of the thesis, I have made additional contributions which are not included in this manuscript.

**Performative Prediction for Classification** This work studies the setting where the distribution of the learning task changes under the performative effects of the model's predictions. Performative learning extends the classical risk minimization framework in that it allows the data distribution to depend on the deployed model. We study the special case of classification and make connections with robustness via a push-forward formula of the distribution shift. This work was published in [Cyf+24].

**Benchmark on Federated Learning for Healthcare** This work proposes various datasets for the cross-silo setting, corresponding to the important use case where a few hospitals collaborate to improve the predictive value of their models. The benchmark aims to be more realistic than existing ones by focusing on true healthcare data and real heterogeneity between centers. This work was published in [Ter+22].

**Course on Dimension Reduction** I designed and taught a 24-lectures in the Master of Machine Learning of Lille University on Dimension Reduction tutorial, with mathematical background and Python implementation, focusing on PCA and then moving on various methods: LLE, t-SNE, spectral embedding, Kohonen map, VAE, LDA.

## 1.6 Publications

During my PhD thesis, I had the opportunity to conduct several research projects with different collaborators, including my supervisor, but also other PhD students and researchers. These projects led to several publications, which are listed below. Each published paper comes with a public code repository.

### Publications in international conferences with proceedings

- Edwige Cyffers and Aurélien Bellet. "Privacy amplification by decentralization". In: *International Conference on Artificial Intelligence and Statistics* (*AISTATS*). PMLR. 2022, pp. 5334–5353
  **Code:** https://github.com/totilas/privacy-amplification-by-decentralization
  **Contribution:** I am the main author.

- Edwige Cyffers et al. "Muffliato: Peer-to-peer privacy amplification for decentralized optimization and averaging". In: *Advances in Neural Information Processing Systems* (*NeurIPS*) 35 (2022), pp. 15889–15902
  **Code:** https://github.com/totilas/muffliato
  **Contribution:** I am a co-first author. I met Mathieu Even, another PhD student from Inria Paris whom I met at NeurIPS in Paris 2021[2]. Mathieu specializes in optimization and I asked him to help with the analysis of gossip algorithms in Network Differential Privacy. Mathieu had the idea to explore multiple gossip steps between gradient steps and suggested moving away from closed-form solutions to directly tackling the sensitivity analysis, leading to the final formulas. I developed the concept of Pairwise Network Differential Privacy, conducted the experiments, contributed to the writing, (and came up with the name *Muffliato*, though Mathieu's support was crucial for its final acceptance).

- Jean Ogier du Terrail et al. "Flamby: Datasets and benchmarks for cross-silo federated learning in realistic healthcare settings". In: *Advances in Neural Information Processing Systems* (*NeurIPS*) 35 (2022), pp. 5315–5334
  **Code:** https://github.com/owkin/FLamby
  **Contribution:** Flamby is a benchmark for cross-silo Federated Learning with natural partitioning, currently focused in healthcare applications, and it was a large collaborative project led by the company Owkin and involving more than 20 people. I coded the first version of the synthetic dataset generation with Paul Mangold (PhD student at Inria Lille). I contributed to the theoretical design of the heterogeneity measures with Paul Mangold and Constantin Philippenko (PhD student at Polytechnique).

- Edwige Cyffers, Aurélien Bellet, and Debabrota Basu. "From noisy fixed-point iterations to private ADMM for centralized and federated learning". In: *International Conference on Machine Learning* (*ICML*) (2023), pp. 6683–6711
  **Code:** https://github.com/totilas/padadmm
  **Contribution:** I am the main author. The idea for the paper came from Debabrota Basu (researcher at Inria Lille), who identified opportunities to extend the privacy amplification results developed in my first paper [CB22] to the ADMM algorithm. The convergence proof is largely due to his expertise in inequalities. The rest of the contributions, including the algorithm, transformation to fixed-point, and experiments, are mine.

- Edwige Cyffers, Aurélien Bellet, and Jalaj Upadhyay. "Differentially Private Decentralized Learning with Random Walks". In: *International Conference on Machine Learning* (*ICML*) (2024)
  **Code:** https://github.com/totilas/DPrandomwalk

[2]NeurIPS in Paris is a local meetup held just before the NeurIPS conference. I presented Muffliato the following year and joined the organizing committee afterwards. Website: https://neuripsinparis.github.io/neurips2024paris/

**Contribution:** I am the main author. Jalaj Upadhyay (Assistant Professor at Rutgers University, whom I had met during an internship before starting my PhD) contributed by deriving examples on specific graphs and helped to the writing.

- Abdellah El Mrini, Edwige Cyffers, and Aurélien Bellet. "Privacy Attacks in Decentralized Learning". In: *International Conference on Machine Learning (ICML)* (2024)
  **Code:** https://github.com/AbdellahElmrini/decAttack
  **Contribution:** I co-supervised Abdellah El Mrini for his Master's thesis. I proposed the project.

- Edwige Cyffers et al. "Optimal Classification under Performative Distribution Shift". In: *Advances in Neural Information Processing Systems (NeurIPS)* 37 (2024)
  **Code:** https://github.com/totilas/PerOptimClassif
  I am the main author. I obtained the main results with Olivier (researcher at the Computer Science Department of ENS). Olivier independently found the variance reduction example. Muni (Fellow in Artificial Intelligence at Université PSL) and Jamal (Professor at the University of Paris-Dauphine) contributed significantly to the connection with robustness. The code is mine.

# Chapter 2

# Decentralized Learning

This chapter provides background in machine learning before moving to the specificities of federated and decentralized learning. We first present the main idea behind supervised learning (Section 2.1). Then, we focus on convex analysis, as the optimization results of this thesis will be established mainly in this framework (Section 2.2). While classical machine learning considers the setting where all the data is centralized, we study decentralized learning, which adds a layer of complexity to the optimization process. We introduce federated learning (Section 2.3), and then move to fully decentralized learning (Section 2.4), which is the central focus of this thesis, where both data and communications are decentralized.

## 2.1 Supervised Learning

> Les statistiques sont des moyens parmi d'autres pour affronter l'inquiétude que suscitent les masses sans formes, c'est-à-dire le chaos[a]. [Des88]
>
> [a] Statistics are one among other means to confront the anxiety that formless masses provoke, that is, chaos.

Machine learning can be introduced as the abstract concept of learning without explicit rules but only concrete examples. This approach assumes that there are underlying mechanisms able to explain the main characteristics of all the examples observed, even if we are not able to code them in a traditional algorithm [GBC16; M.16]. This assumption is extremely strong as it suggests the existence of an underlying probability distribution that could be, at least to some extent, captured by tuning parameters on only a finite number of examples that cannot cover the entire space of possible inputs. The hypothesis of machine learning is still controversial today, even in science: it is common to hear doctors claim that each patient is unique, and to refuse to estimate a probability of success for a given treatment. Relying on global trends and correlation to learn rather than on a theory is an audacious gamble.

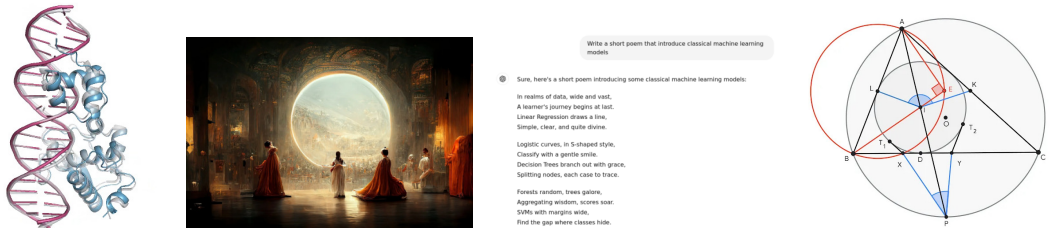



**Figure 2.1** – Example of machine learning applications. Alphafold 3 (2024) predictions are close the grey ground truth for DNA protein binding, Théâtre D'opéra Spatial (2022) generated by Midjourney and prompted by Jason Michael Allen won an art prize, ChatGPT (2022) generate plausible conversation taking advantage of encyclopedic knowledge in every domain. AlphaProof and AlphaGeometry reach silver medal level at IMO 2024, solving the geometry problem in 19 seconds.

Historically, Desrosières described the long debate during the nineteenth and twentieth century on the added value of computational methods and subsequently statistics [Des14], where statisticians faced skepticism and misunderstanding. In this thesis, we however embrace the faith in machine learning and assume that all unique examples reflect a universal phenomenon that mathematics can not only capture, but also predict.

This faith is supported by the recent successes of deep learning: Large Language Models such as the (in)famous ChatGPT, demonstrate the ability to enforce grammar and logical thinking to some extent without formalizing the grammar of natural languages. Most strategy games, such as chess [Sil+17], go, and video games[Vin+19] are better solved by machine learning models than humans. Diffusion models[HJA20] generate images with extremely realistic features, and deep learning predicts with more accuracy various physical phenomena than fully explicit mathematical models can, and even problems of maths Olympiad [At24] are now in the scope of specialized models (Figure 2.1).

Machine learning problems are often divided into unsupervised learning, supervised learning, and reinforcement learning. These three branches do rely on learning from the data, but interact with it differently: unsupervised learning directly exploits the structure of the distribution of data to find patterns, while supervised learning requires labeled data and reinforcement learning collects data from its own interaction with the environment that produces the data. We focus here on supervised learning [Bac24].

Labeled data in supervised learning are presented as a collection of pairs $(x_i, y_i)_{1\leq i\leq n} \in (\mathcal{X} \times \mathcal{Y})^n$ where $x \in \mathcal{X}$ is an input and $y \in \mathcal{Y}$ the output the machine learning model should learn to predict. $\mathcal{X}$ can describe various domains such as text documents, pictures, sounds, tabular data with sensor measurements, or time series. Despite this semantic heterogeneity, we assume it is possible to encode this input as a vector $x \in \mathbb{R}^d$, which means, of course, that $d$ might need to be large. The output $y \in \mathcal{Y}$, also called the label, could also be arbitrary, but we

often restrict ourselves to regression tasks with real values ($y \in \mathbb{R}$) or to classification tasks where $y \in \{1, \dots, k\}$ or even $y \in \{0, 1\}$ for binary classification.

It is assumed that all the data points belonging to the training set $(x_i, y_i)$ can be seen as random variables that are independent and with the same distribution as the ones that the model should be able to predict at testing time. Coping with non independent samples, such as in time series, or with distribution shifts, are important research topics, but will not be covered in this thesis. We thus denote $p$ the probability distribution of the samples.

In order to learn adequately, we should evaluate whether the predictions made by the model are close to the outputs it should produce. This is measured through a loss function $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$, that takes the prediction of the model $z$ and the output $y$ and returns a positive value that grows with the magnitude of the error in the prediction. For example, in binary classification, a natural loss is the $0 - 1$ loss, defined by $l(y, z) = \mathbf{1}_{y \neq z}$. However, it is often convenient for training to consider other loss functions with better properties (see Section 2.2).

Following up on the introductory story (Section 1.1), a possible fictive dataset ShopData (Table 2.1) could contain the goods bought by various customers along with the date of transaction and the price. From this, one can deduce a classification task, such as whether a customer will buy a new good or not, which is clearly an interesting prediction for optimizing stocks and profits for supermarket owners. One could also deduce a regression task with the same dataset, by predicting the price of a given item. Internally, the model should capture similar features to solve both regression and classification tasks. For classification, a usable version is likely to use one-hot encoding for all products, and returns only one row per shopping basket of a given customer on a given day, where all bought items are marked with the quantity taken, and other items have zero. It is thus a high-dimensional dataset as it has the dimension of the number of products in the store. This is usual in machine learning: for instance, a small image dataset such as MNIST has a dimension $d = 784$ and language tasks need even bigger dimensions.

**Table 2.1** – Example of fictive dataset ShopData inspired by the shopping basket problem in Section 1.1.

| Client ID | Date | Item | Price (€) |
|---|---|---|---|
| 501-AXT | 2024-08-01 | Apple | 0.45 |
| 202-BYU@E | 2024-08-01 | Bread | 1.10 |
| 501-AXT | 2024-08-02 | Sanitary Product | 5.50 |
| 303C-XYZA04 | 2024-08-02 | Milk | 0.89 |
| 404D-123-AB | 2024-08-03 | Cheese | 2.50 |
| 202-BYU@E | 2024-08-03 | Apple | 0.40 |
| 005-E89-76T | 2024-08-04 | Biscuits | 1.30 |
| 303C-XYZA04 | 2024-08-04 | Orange Juice | 1.20 |

Let us denote by $\mathcal{A}$ the trained algorithm, so that the predicted output for the input $x$ is given by $\mathcal{A}(x)$. The goal is to minimize the expected risk:

$$\mathcal{R}(\mathcal{A}) = \mathbb{E}[l(y, \mathcal{A}(x))] = \int_{x \times y} l(y, \mathcal{A}(x)) dp(x, y)$$

which depends on the unknown distribution $p$. Hence, we would like to compute $\mathcal{R}$ but we cannot integrate on $p$, we thus use the empirical risk instead, computed on our samples $(x_i, y_i)_{1 \leq i \leq n} \in (\mathcal{X} \times \mathcal{Y})^n$.

**Definition 2.1** (Empirical Risk)**.** *The empirical risk $\widehat{\mathcal{R}}$ of an algorithm $\mathcal{A}$ is defined by:*

$$\widehat{\mathcal{R}}(\mathcal{A}) = \frac{1}{n} \sum_{i=1}^{n} l\left(y_i, \mathcal{A}\left(x_i\right)\right)$$

This empirical approach, however, introduces the risk of memorizing the training set and learning a model that does not perform well at testing time. This is the risk of overfitting. Mitigating this problem is part of the difficulty of training models, and the theory does not always correctly describe this phenomenon: empirically, deep learning exhibits a double descent [Nak+19], which means that conventional statistical learning theory that predicts overfitting fails to accurately capture the dynamics of generalization that happens in machine learning models. As translating the value of the loss function into predictive power is difficult, we always report not only the decrease of this loss function but also performance on a testing set that is not used during the training.

Behind the generic formulation of an algorithm $\mathcal{A}$, we refer in practice to a parametrized family of prediction functions, where we learn a parameter $\theta \in \Theta$ that typically lives in a subset of a vector space. Denoting by $f_\theta$ the model resulting from the choice of parameter $\theta$, we aim at finding $\hat{\theta}$ that minimizes the empirical risk.

**Definition 2.2** (Empirical Risk Minimization)**.** *The empirical risk minimization consists in finding $\hat{\theta}$ such that:*

$$\hat{\theta} \in \underset{\theta \in \Theta}{\arg\min}\, \widehat{\mathcal{R}}\left(f_\theta\right) = \frac{1}{n} \sum_{i=1}^{n} l\left(y_i, f_\theta\left(x_i\right)\right) \tag{2.1}$$

## 2.2 Elements of Convex Optimization and Gradient Descent

> The great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity. [Roc93]

### 2.2.1 Regularity assumptions

Solving Equation (2.1) can be arbitrarily hard in the general case, as the space $\Theta$ is too big to be easily explored. In particular, training a machine learning model generally consists in iteratively and slowly updating the parameters to steer the model to a better place. Such approaches require some regularity assumptions if one wants theoretical guarantees that the training will converge to an optimal model. In this section, we present assumptions that will be used in the thesis when deriving guarantees, which mainly rely on convexity. These assumptions are not always satisfied by deep neural networks, however they can be satisfied by simpler machine learning models, and the theory developed under this framework can be partially extended to other settings. The privacy guarantees will usually not require convexity but bounded sensitivity, informally corresponding to the bound on how much a function varies on different inputs (see Section 3.2).

**Definition 2.3** (Convexity)**.** *A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if for all points $u, v \in \mathbb{R}^d$ and scalar $\lambda \in (0, 1)$,*

$$f(\lambda u + (1 - \lambda)v) \leq \lambda f(u) + (1 - \lambda) f(v)$$

Note that convex functions have the nice property that a local minimum is a global minimum. However, these functions can have multiple minima or no minimum at all. We hence often need to go further in this direction by imposing strong convexity, which ensures that the minimum exists and is unique.

**Definition 2.4** (Strongly convexity)**.** *For $\mu > 0$, $f$ is $\mu$-strongly convex if the function $f - \frac{\mu}{2}\|\cdot\|^2$ is convex.*

Regularity assumptions are quite important for optimization methods used in machine learning; gradient descent requires the function to be differentiable, and we sometimes also need regularity of the gradient with twice differentiable functions. Convexity can be nicely formulated as "being above its chords", which corresponds to the following inequality for a differentiable function:

$$\forall u, v \in \mathbb{R}^d, f(u) \geq f(v) + \langle \nabla f(v), u - v \rangle$$

Finally, convexity for a twice differentiable function is equivalent to $\nabla^2 f(u)$ being positive semi-definite for all $u$, where the matrix $\nabla^2 f$ is the Hessian of $f$. Strong convexity is equivalent to $\nabla^2$ being positive definite.

To ensure that a function does not change too quickly, and in particular, to bound the sensitivity, a useful property is Lipschitz continuity.

**Definition 2.5** (L-Lipschitzness). *For $L > 0$, a function $f$ is $L$-Lipschitz if for all points $u, v \in \mathbb{R}^d$, the following equality holds*[a]

$$\|f(u) - f(v)\| \leq L \|u - v\|$$

[a] Note that despite the fact we note the two norms with the same notation, they are not identical in general.

As we will often directly work with gradients, the same property transposed to the first derivative is useful.

**Definition 2.6** ($\beta$-smoothness). *For $\beta > 0$, a differentiable function $f$ is $\beta$-smooth if its gradient is $\beta$-Lipschitz.*

In order to have bounded sensitivity, it is often needed in practice to preprocess the data. A usual technique is to standardize the features, which means that we enforce them to have zero mean and unit variance, so each feature has similar importance a priori before starting to train the model.

Once $L$-smooth and $\mu$-strongly convex, a function is easily monitored by the two following two inequalities:

$$\forall u, v \in \mathbb{R}^d, \quad \frac{\mu}{2}\|u - v\|^2 \leq f(u) - f(v) - \langle \nabla f(v), u - v \rangle \leq \frac{\beta}{2}\|u - v\|^2 \tag{2.2}$$

And when $f$ is also twice differentiable, we have:

$$\mu I_d \preceq \nabla^2 f \preceq \beta I_d$$

The ratio $\kappa = \beta/\mu$ is called the *condition number* and often appears in convergence results. The bigger it is, the slower the convergence will be. This can be thought as a way to tame the curve between two quadratics, as seen in Figure 2.2.

Taking again the classification problem associated to ShopData, the first loss to come up is the $0 - 1$ loss that puts a penalty of $1$ for any misclassification. However, the loss has a discontinuity and does not satisfy the previous properties. In practice, we will use a *surrogate loss* with better training properties, such as the logistic loss. For convenience, the two labels are $-1$ and $1$, which ensures that each class corresponds to a sign. The logistic loss is then $l(y, z) = \log(1 + \exp(-yz))$. In this setting, to minimize the risk, the predictions should have the correct sign and their absolute value should grow with the confidence of the model.

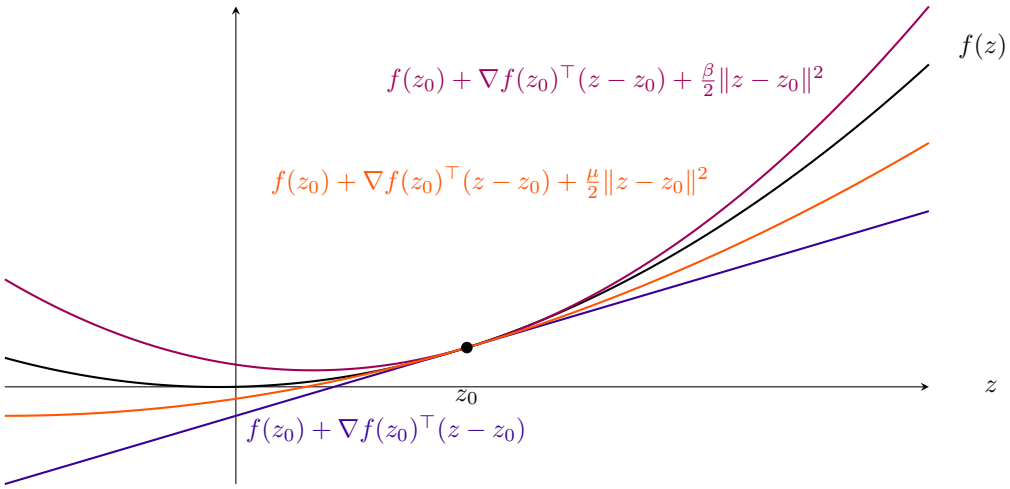



**Figure 2.2** – Quadratic upper bound and lower bound provided respectively by $\beta$-smoothness (in magenta) and $\mu$-strong-convexity (in orange) at a point $z_0$ for a function $f$.

The training consists of finding a good parameter $\theta \in \mathbb{R}^{d+1}$ for $f_\theta$. [1] The associated model is simply given by the scalar product $x^T\theta$, and thus the learning task is reduced to:

$$\hat{\theta} \in \arg\min_{\theta \in \Theta} \quad \sum_{i=1}^{n} (\log(1 + \exp(-y_i \theta^\top x_i)))$$

### 2.2.2 First-order optimization

After surveying these properties on functions, we introduce how to find their minimum, we aim to find a point $u$ that minimizes a function $f$. The most well-known first-order method is gradient descent, which eventually finds a minimum by moving at each step in the direction of the steepest descent, as given by the gradient. From an initial point $u_0$, the sequence is defined by the following iterations

$$u_{t+1} = u_t - \gamma_t \nabla f(u_t) \quad \text{for } t \geq 0 \quad ,$$

with $\gamma_t$ being the sequence of step sizes. We note $\gamma = \gamma_t$ when the sequence $(\gamma_t)$ is constant. This algorithm was first introduced in 1847 [Cau+47], and benefits from a large literature on its convergence under various conditions [Nes13], and especially in the convex setting [BV04]. As an illustration, for the simplest case of a $\mu$-strongly convex and $L$-smooth function, we have the following result.

[1] In practice, to learn an intercept, we add a dimension where all the inputs are equal to one.

**Theorem 2.7** (Convergence of GD [Nes13]). *If $f$ is $\mu$-strongly convex and $L$-smooth, and if there exists a global minimum $u_*$, gradient descent with constant step size $\gamma = 1/L$ satisfies*

$$f(u_t) - f(u_*) \leq \left(1 - \kappa^{-1}\right)^t (f(u_0) - f(u_*))$$

*Taking $\gamma = 2/(\mu + L)$ leads to the stronger result*

$$f(u_t) - f(u_*) \leq \left(1 - \frac{2}{\kappa + 1}\right)^{2t} \frac{L}{2} \|u_0 - u_*\|^2$$

In the case of machine learning, computing the full gradient is, however, very costly: even the small MNIST dataset has $70,000$ data points. Instead of computing an exact gradient, it is possible to use an unbiased estimator, at the price of introducing a variance term. The estimator is typically the gradient with respect to a unique point or a small batch of points. Stochastic Gradient Descent (SGD) then corresponds to the following iterative algorithm:

$$u_{t+1} = u_t - \gamma_t g_t(u_t) \text{ with } \mathbb{E}(g_t|\mathcal{F}_t) = \nabla f(u_t)$$

where $\mathcal{F}_t$ is the filtration at time $t$, informally corresponding to the subset of all previously drawn variables[2]. In this case, the iterates are stochastic; hence the convergence results are also stochastic and depend on the variance of the estimates. One can argue that stochastic gradient descent is ill-named, as it is not a descent algorithm: the iterates are not monotonic, and some updates might increase the loss.

In practice, the choice of the small sample of the data – a batch or mini-batch – is done randomly or by cycling over the whole dataset. To make the dependency explicit, we can write $g_t^B$ for a computation over the batch $B$. Another way to express the sample dependency is to consider a loss function with two arguments – the parameter and the sample – often just changing between upper case and lower case, then $g_t = \nabla_u F(u_t, \xi_t)$ where $f(u) = \mathbb{E}_{\xi \sim D}(F(u, \xi))$.

**Theorem 2.8** (Convergence of SGD [Gow+19]). *If $f$ is $\mu$-strongly convex and $\beta$-smooth, if there exists a global minimum $u_*$, if $g_t(u_t)$ satisfies $\mathbb{E}(g_t(u_t)|\mathcal{F}_t) = \nabla f(u_t)$, $\mathbb{V}(g_t(u_t)|\mathcal{F}_t) \leq \rho^2$ and $\gamma < 1/2\beta$, then the following holds for stochastic gradient descent:*

$$\mathbb{E}\left(\|u_t - u_*\|^2\right) \leq (1 - \gamma\mu)^t \|u_0 - u_*\|^2 + \frac{2\gamma\rho^2}{\mu}$$

We see that this result is similar to the previous one but it is now a result in expectation and it includes a second term that increases with the level of noise of the estimates. In practice, this

[2]The filtration will be omitted when the context is clear.

means that if one wants to achieve a precision $\iota$, fixing $\gamma = \min(\frac{\iota\mu}{4\rho^2}, \frac{1}{2\beta})$ gives this precision after $\max(\frac{4\rho^2}{\iota\mu^2}, \frac{2\beta}{\mu}) \log(2\|u_0 - u_*\|^2/\iota)$ steps. It is possible to refine these results significantly (see for instance [GG23]), but the same main ideas remain: the condition number is a good proxy for the hardness of the problem, and the noisier the gradient, the larger the ball within which we converge is. In fact, some analyses show that the noise that really matters is the noise at the optimal point, i.e., we can derive guarantees that depend on $\mathbb{E}\left[\|g_t(u_t) - \nabla f(u_*)\|^2\right]$. Theorem 2.8 is strong because is characterized the convergence of the iterates to the optimal point. When convexity is not assumed, this is often impossible, and convergence results can be expressed as inequalities on the norm of gradients iterates, or on the average of iterates [Nes13; GG23].

## 2.3 Federated Learning

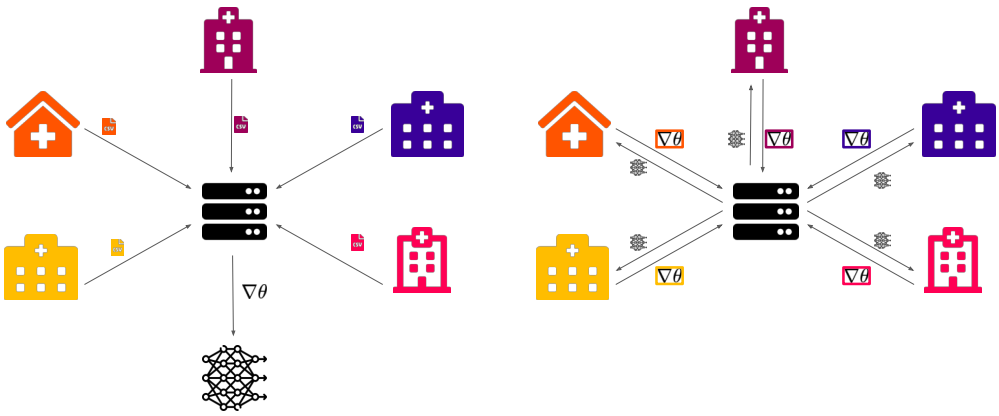
> FL embodies the principles of focused data collection and minimization, and can mitigate many of the systemic privacy risks and costs resulting from traditional, centralized machine learning and data science approaches. [Kai+21]

In the previous sections, we introduced empirical risk minimization and stochastic gradient descent without discussing the availability of the data during the computation. We thus implicitly assumed a centralized setting, where a single entity has all the data and can access it easily. This entity, whether a company, an abstract cloud, or a trusted curator, might not exist in some practical use-cases. We might even want to avoid such an entity for privacy reasons, and we defer the reader to Chapter 3 for these aspects. Letting aside privacy reasons, the collection of the data is often decentralized by design: sensors that are in different locations, health data coming from various hospitals, logs from smartphones, purchases in various places.

Centralizing raw data has a cost, not only for data safety, but also in terms of storage, transmission costs, and latency implied for data gathering, and it limits scalability by the capacity of the central entity [Li+20; Kai+21]. Avoiding data centralization is thus a promising direction, especially as machine learning moves towards processing larger quantities of data, which are more sensitive and complex.

Federated learning [McM+17] aims at training a model collaboratively while keeping data decentralized (see Figure 2.3). The participants are typically referred to as users, centers, devices, clients or individuals, and we will denote by $n$ the number of participants. The dataset $\mathcal{D}$ used for training the model is the union of each local dataset $\mathcal{D}_u$ belonging to user $u$: $\mathcal{D} = \cup_{i=1}^{n} \mathcal{D}_i$. Each dataset might have a different size and different underlying distribution.

For simplicity, in the whole thesis we assume that we only want to construct a single global model parametrized by $\theta$ which means that we let aside the interesting problems related to




(a) Centralized Learning (b) Federated Learning

**Figure 2.3** – Comparison between centralized and federated learning

personalization [Smi+17]. The optimal model is the one that minimizes the average of the local losses $f_i$:

$$\theta_* \in \arg\min_\theta \frac{1}{n}\sum_{i=1}^{n} f_i(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{\xi_i \sim \mathcal{D}_i}\left(F_i(\theta, \xi_i)\right) \tag{2.3}$$

Note that the local loss is minimized empirically on the local dataset as discussed previously (Equation (2.1)). The average can be weighted by the size of the local dataset or any other relevant weighting schemes. For instance, in the case of logistic regression, the local loss can be written as follows:

$$f_u(\theta) = \frac{1}{|\mathcal{D}_u|}\sum_{(x,y)\in\mathcal{D}_u} \log\left(1 + \exp\left(-y\theta^\top x\right)\right)$$

To solve the optimization problem in a federated setting, the approach originally proposed by [McM+17] is the FedAvg algorithm, shown in Algorithm 2.1. At each round, some users are selected at random, receive the current model and compute several gradient steps on their local datasets before sending back the updated version to the server. The new global model is obtained from the average of these local updates. From this general framework, it is possible to improve the performance by accommodating the limited availability of devices, tackling asynchronous updates, dropout, difference in data quality... Refinements can come from the sampling procedure [Fra+21], the nature of the local updates [Kar+20], the way to average the local models [Jhu+22; Red+20], the potential compression used for communication [SGSJ21; Had+21].

Federated learning requires tackling a lot of difficulties inherent to this setting, the first one being data heterogeneity, which can come from various origins such as variation in the data collection (not the same scanner in different hospitals), differences in population (a city in a wealthy neighborhood has a global shift in life expectancy), or even specialized centers

(a center specialized in a given disease has much more labeled data than other ones). This raises the question of personalizing the model for each center or for a combination of centers rather than sticking to a global one. A second challenge is center heterogeneity, where centers can have various levels of computing resources, different memory size, different bandwidth, and different latency, which requires algorithms to be robust to the different kinds of centers encountered, while keeping the communication cost low and efficiency high enough. In these regimes, being able to detect potential malicious centers that could poison their data, or even detect dysfunctional centers, remaining stable and being able to interpret the results is still challenging without seeing the data. These challenges will not be studied in this thesis.

**Algorithm 2.1:** Federated Averaging (FedAvg)

**Input:** Initial model $\theta_0$, number of communication rounds $T$, local epochs $K$, learning rate $\gamma$, batch size $b$, and the proportion $p$ of active workers at each round
**Output:** Global model $\theta_T$
**for** $t = 1, 2, \dots, T$ **do**
  $S_t \leftarrow$ random set of $\lceil pn \rceil$ clients
  **foreach** *client* $i$ *in* $S_t$ **do**
    $\theta_i \leftarrow \theta_{t-1}$
    **for** $k = 1, 2, \dots, K$ **do**
      $\xi_{k,i} \leftarrow$ Random sample from $\mathcal{D}_i$ of size $b$
      $\theta_i \leftarrow \theta_i - \gamma \nabla_\theta f(\theta_i, \xi_{k,i})$
    $\theta_{i,t} \leftarrow \theta_i$
  $\theta_t \leftarrow \frac{1}{n} \sum_{i=1}^{n} \theta_{i,t}$

Training and deploying federated learning systems has enjoyed some success stories, such as in the Apple ecosystem [Pau+22] or the Google one [Bon+19] in the cross-device setting and for hospital consortia [Rie+20] in the cross-silo setting.

## 2.4 Decentralized learning

Federated learning with a central server still requires a quasi-omniscient server, able to communicate with all the devices, to be trustworthy, to remain present during the whole training and to support all the communication costs. This is thus a bottleneck to scaling up the learning [Kai+21], especially for the cross-device setting. In Chapter 3, we will see that the central server also acquires lots of information on the local datasets even if it does not access the local raw data. It is possible to remove this central entity and still collaborate by relying only on peer-to-peer communication [Lia+17; Neg+19; Neg+20]: we call this setting decentralized learning[3] and this is the setting that will be studied in this thesis.

[3] This setting can also be referred to as Fully Decentralized Learning to emphasize the absence of a central server.

We call *communication graph* the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ which has the $n = |\mathcal{V}|$ users as nodes and where an edge $(u, v) \in \mathcal{E}$ indicates the possibility to exchange messages between $u$ and $v$. The goal is still to solve the sum of the local loss functions as defined in Equation (2.3), but with only peer-to-peer communications. As a warm-up, we first focus on computing a simple average (Section 2.4.1) before moving on Decentralized Stochastic Gradient Descent (Section 2.4.2) and conclude with discussing the possible choices of communication graph (Section 2.4.3).

### 2.4.1 Average computation

In order to aggregate the local models into a central one similarly to the aggregation step in FedAvg, we need a procedure that computes global aggregations while relying only on local communication along the edges of the graph. We thus first present how to compute the arithmetic average of local values over the graph. Starting from the problem of averaging before going to optimization is an approach that will be often used in this thesis. We will denote as $x_u$ the value of the node $u$.

Given a graph $\mathcal{G}$, we can construct a gossip matrix that appropriately weights the local contributions of the nodes.

**Definition 2.9** (Gossip matrix)**.** *A gossip matrix over a graph $\mathcal{G}$ is a matrix $W \in [0,1]^{n\times n}$ with non-negative entries, symmetric, aperiodic, irreducible, doubly stochastic – i.e $W\mathbb{1} = \mathbb{1}$ and $\mathbb{1}^\top W = \mathbb{1}^\top$ – and such that for any $u, v \in \mathcal{V}$, $W_{uv} > 0$ implies that $\{u, v\} \in \mathcal{E}$ or $u = v$.*

We adopt this very complete definition as it ensures that all the results presented in the next chapters hold. However, it is often possible to remove several assumptions, and we discuss the consequences below.

Assuming that $W$ is symmetric implies that each edge can be traversed in both directions, with the same probability. In particular, this assumption corresponds to only considering undirected graphs. Ensuring the symmetry of $W$ has the significant advantage of enabling the use of the spectral theorem, which proves useful in Chapter 7. However, most of the other results do not rely on this assumption.

The last constraint on positive coefficients ensures that messages are only sent where they can transit. Stochasticity ensures that the total mass over the graph is conserved across iterations. These two hypothesis are thus crucial to any learning process. We also assume that the gossip matrix is aperiodic and irreducible, i.e., there exists a time $t_0$ such that for all $t \geq t_0$ and any pair of vertices $u$ and $v$, $W^t_{uv} > 0$. This ensures that, as long as we consider a number of steps $t$ large enough, there exists a path from $u$ to $v$ in exactly $t$ steps. Under these assumptions, there

exists an *invariant distribution* $\pi$ for $W$:

$$\pi = \pi W$$

For any connected graph, we can easily construct a gossip matrix such that the invariant distribution is the uniform distribution $\pi = \mathbb{1}/n$, giving the same weights to every participant in the final model. In this case, the gossip matrix is *doubly stochastic*. It corresponds to what is often done in practice, as the uniform weights makes sense in a lot of situations. The Hamilton weighting satisfies this condition by putting weights uniformly among the neighbors. Let $d_u$ denote the degree of node $u$. Then:

$$W_{uv}^{\text{ham}} = \frac{1}{\max\{d_u, d_v\}},$$
$$W_{uu}^{\text{ham}} = 1 - \sum_{v \neq u} W_{uv}^{\text{ham}}.$$

This weighting gives a symmetric gossip matrix, which in particular ensures the intuitive property that each edge is used in both directions. Assuming the gossip matrix is symmetric enables the use of useful linear algebra properties, as the spectral theorem can be applied. Note that one could take the sum instead of the maximum without changing the properties of the resulting matrix.

**Gossip** Given a gossip matrix, we can define *Synchronous Gossip* as the algorithm executed locally and iteratively in order to compute a global average on the graph, as depicted in Figure 2.4. Starting from an initial set of values $(x_{v,0})_{v \in \mathcal{V}}$, at each step, each node receives the local values of its neighbors and aggregates them:

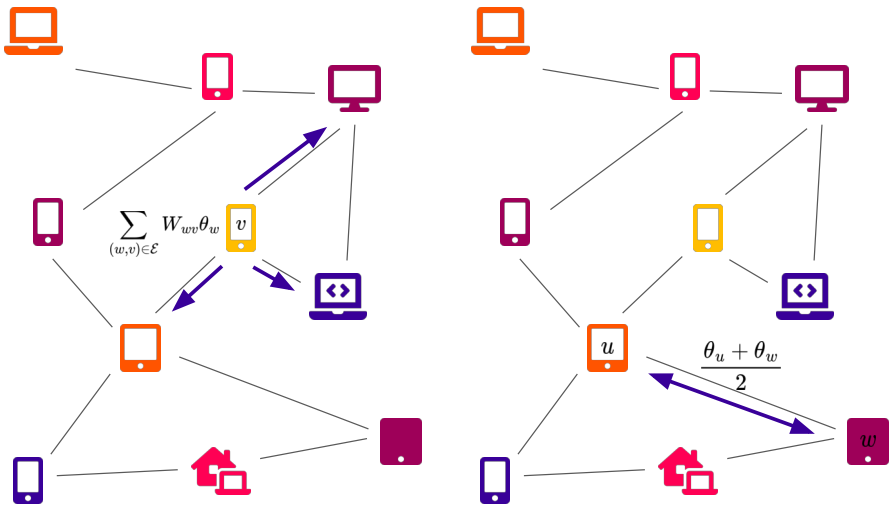
$$x_{v,t+1} = \sum_{w \in \mathcal{N}_v} W_{v,w} x_{w,t}$$

Let $X$ be the stacked value $x_v$ over the nodes, the iterates can be concisely written as a matrix multiplication:

$$X_{t+1} = W X_t$$

By iterating this from the initial local values, all the local values converge to the weighted average $\sum_{v \in \mathcal{V}} \pi_v x_v$. The convergence rate depends on the spectral gap, that is the difference between the largest eigenvalue – it is always 1 by construction with our gossip matrix definition – and the second one.

**Definition 2.10** (Spectral gap)**.** *The spectral gap $\lambda_W$ associated with a gossip matrix $W$ is $\min_{\lambda \in Sp(W) \setminus \{1\}}(1 - |\lambda|)$, where $Sp(W)$ is the spectrum of $W$.*




(a) Synchronous Gossip (b) Asynchronous Gossip

**Figure 2.4** – In synchronous gossip, all nodes perform synchronous updates: they aggregate their private value with their neighbors and then share it again, while in asynchronous gossip, only one edge is active at each time step.

The iterates $x_{t+1} = Wx_t$ converge to the mean with a linear rate $e^{-t\lambda_W}$ [BV04]. This convergence rate can be accelerated to $e^{-t\sqrt{\lambda_W}}$ by using Chebychev polynomials [BBG20], by replacing the power of $W$ by these polynomials, which can also be computed in a decentralized fashion. This trick is used in Chapter 6.

**Definition 2.11** (Re-scaled Chebychev polynomials)**.** *The re-scaled Chebychev polynomials* $(P_t)_{t\geq 0}$ *with scale parameter* $\gamma \in [1, 2]$ *are defined by second-order linear recursion:*

$$P_0(X) = 1\,, \quad P_1(X) = X\,, \quad P_{t+1}(X) = \gamma X P_t(X) + (1-\gamma)P_{t-1}(X)\,,\, t \geq 2\,. \tag{2.4}$$

Note that for the complete graph, the gossip matrix is $\mathbb{1}\mathbb{1}^T/n$, so the second eigenvalue is $0$ and the convergence is achieved after a single iteration. It is possible to obtain the asymptotic number of iterations for other well-known graphs, as reported in Table 2.2.

Synchronous gossip is not the only way to propagate information in the graph. An alternative is *Asynchronous Gossip*, also called *Randomized Gossip* [Boy+06], where at each step a unique edge wakes up independently from the previous steps, and the nodes at its endpoints average their values (see Figure 2.4). The probability of each edge waking up is then proportional to the values in the gossip matrix to ensure the same convergence properties as before. In this case, we can still express the iterates as matrix multiplication, which now requires to consider a sequence $(W_t)$ of random matrices, where for activating the edge $(u, v)$ we use

$$W_{\{u,v\}} = I - \frac{(e_u - e_v)\,(e_u - e_v)^\top}{2}$$

**Random walks** An alternative to gossip is random walks, where a token follows a Markov chain on the graph with probability given by a transition matrix [LS07; JRJ10]. Note that the token cannot "fly" without edge from a vertex to another, creating dependency relationships between the activation of edges, unlike in asynchronous gossip. For simplicity, we will use a gossip matrix as transition matrix, as defined in Definition 2.9. For convergence analysis, it is sometimes useful to consider how quickly the probability of the token being at a given node approaches the invariant distribution $\pi$. This can be quantified by the mixing time.

**Definition 2.12** (Mixing time)**.** *The mixing time* $\tau_{\text{mix}}(\iota)$ *of a Markov chain of gossip matrix* $W$ *is the time needed for the random walk to be close to a factor* $\iota$ *of its asymptotic behavior:*

$$\tau_{\text{mix}}(\iota) = \min\left(t : \max_v \left\|(W^t)v - \pi\right\|_{TV} \leq \iota\right)\,, \tag{2.5}$$

*where* $\|P - Q\|_{TV}$ *is the* total variation distance *between two probability measures* $P$ *and* $Q$ *defined over the same measurable space* $(\Omega, \mathcal{F})$, $\|P - Q\|_{TV} = \sup_{A\in\mathcal{F}} |P(A) - Q(A)|$.

Then, the average of all the local values can be computed by a running average as the token goes from one node to another.

### 2.4.2 Decentralized SGD

From the propagation mechanisms defined above, we can build more complex algorithms to solve optimization tasks and in particular minimizing the sum of local loss functions (Section 2.2 and Equation (2.3)). To do so, spreading information steps should alternate with computing local optimization phases. The most well-known algorithm for this is Decentralized Stochastic Gradient Descent (D-SGD) [Lia+17; Kol+20] based on synchronous gossip, as presented in Algorithm 2.2.

**Algorithm 2.2:** Decentralized Stochastic Gradient Descent (D-SGD)

**Input:** Initial model $\theta_0$, number of communication rounds $T$, gossip matrices $(W_t)$, learning rate $\gamma$
**for** $t = 1, 2, \dots, T$ **do**
  **foreach** *node* $v$ *in* $\mathcal{V}$ **do**
    $\xi_{v,t} \leftarrow$ Random sample from $\mathcal{D}_v$
    $\theta_{v,t+1/2} \leftarrow \theta_{t,v} - \gamma \nabla_\theta f(\theta_{v,t}, \xi_{v,t})$
    Send $\theta_{v,t+1/2}$ to nodes $u$ such that $W_{vu,t} > 0$
    Receive $\theta_{u,t+1/2}$ from $\mathcal{N}_v = \{u : W_{uv,t} > 0\}$
    $\theta_{v,t+1} \leftarrow \sum_{u \in \mathcal{N}_v} W_{uv,t} \theta_{u,t+1/2}$
**Output:** Local models $(\theta_{v,T})_v$

Note that even though we choose to present this algorithm by describing the local updates involved, it is still possible and often convenient to think in terms of matrices. By denoting $G_t$ the stacked gradient estimates of all the nodes, D-SGD corresponds to the update rule

$$\theta_{t+1} = W(\theta_t - \gamma G_t)$$

There exists a lot of variant of this algorithm, for instance by doing more gossiping steps between computation, by doing more local steps between gossip, by accelerating the gossip, by using dual methods [Kol+20; MR16; HBM20]. D-SGD benefits from parallelization, as all nodes compute gradients in parallel in contrast with the centralized alternative and avoid the communication bottleneck of federated learning as the gossiping relies on peer-to-peer communication on a suitable graph.

We now present a convergence result for D-SGD that holds under a quite flexible assumption about the sequence of gossip matrices, as it requires only properties over the expected value over several iterations. This assumption allows for possible dropouts, randomized gossip, and various number of local updates. The only constraint is a geometric decay after a given number of iterations $\omega$.

**Assumption 2.13** (Expected Consensus Rate). . *There exists two constants $r \in (0, 1]$ and integer $\omega \geq 1$ such that for all matrices $X \in \mathbb{R}^{d\times n}$ and all integers $k \in \{0, \dots, T/\omega\}$*

$$\mathbb{E}_W \left\| XW_{k,\omega} - \bar{X} \right\|^2 \leq (1-r)\|X - \bar{X}\|^2$$

*where $W_{k,\omega} = W_{(k+1)\omega-1} \cdots W_{k\omega}$ and $\bar{X} = X\frac{\mathbf{1}\mathbf{1}^\top}{n}$ and $\mathbb{E}$ is taken over the distributions $W_t \sim \mathcal{W}^{(t)}$ and indices $t \in \{k\omega, \dots, (k+1)\omega - 1\}$.*

**Theorem 2.14** (Convergence of D-SGD [Kol+20]). *Under Assumption 2.13, for a loss $\mu$ strongly convex and $\beta$-smooth, the iterates of Algorithm 2.2 converge with the following guarantee.*

$$\frac{1}{n}\sum_{i=1}^{n} \|x_{T,i} - x_*\|_2^2 = \tilde{\mathcal{O}}\left( \frac{\rho^2}{n\mu T} + \beta\frac{\omega^2\sigma_*^2 + \omega r\rho^2}{\beta^2 r^2 T^2} + \frac{\beta\omega\|x_0 - x_*\|^2}{r}\exp\left[-\frac{\mu T r}{\beta\omega}\right]\right)$$

*where the gradient noise at the minimizer $x_*$ is assumed to be bounded by $\|\nabla f_v(x_*)\|^2 \leqslant \sigma_*^2$ for all $v \in V$ and $\rho = \sum_{i=1}^n \mathbb{E}_{\xi_i} \|\nabla F_i(x_*, \xi_i) - \nabla f_i(x_*)\|_2^2$.*

Note that, in the special case of using a single deterministic gossip matrix as in Definition 2.9, a possible value for $r$ is given by $1 - r = (1 - \lambda_2)^2$. This is obtained by rewritting $WX - \bar{X} = W(X - \bar{X})$ and applying Perron Frobenius theorem.

The synchronicity of all updates is however costly in practice, as it requires to wait for slower nodes – often called stragglers – and that all nodes remain available during the training procedure. It is possible to use asynchronous gossip instead, but nodes still need to be available during all the training procedure in this case. These limitations lead to prefer random walks in some context. In this case, the unique model is held by the token that walks on the graph, contrary to gossip algorithms where each node maintains and updates its own model. At each step, a gradient is estimated from the local data of the node $v_t$ currently holding the token before moving to one of its neighbors:

$$\theta_{t+1} = \theta_t - \gamma\nabla_\theta f(\theta_t, \xi_{v_t,t}) \tag{2.6}$$

In this scenario, convergence results are more difficult to obtain, as the convergence depends on the specific path taken by the token, and steps are dependent from each other. Bounding how much the updates differ from the uniform sampling over the graph is thus hard and the optimization results are rather recent. A first result was given by [JRJ10] but with guarantees on the minimum of all iterates. Stronger and more general analyses have then been proposed by [Mao+20; Hen22], and we present here only a recent result in the smooth and strongly convex setting.

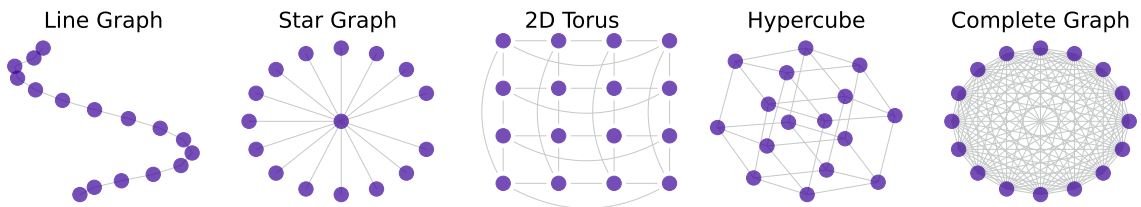



**Figure 2.5** – Example of classical graphs with $n = 16$ nodes

**Theorem 2.15** (Convergence of RW-DSGD [Eve23]). *Let $(f_v)$ be $\beta$-smooth and strongly convex functions, such that the gradient noise at the minimizer $x_*$ can be bounded by $\|\nabla f_v(x_*)\|^2 \leqslant \sigma_*^2$ for all $v \in V$. For a well chosen stepsize $\gamma > 0$, the iterates generated by Equation (2.6) satisfy:*

$$\mathbb{E}\left[\|x_T - x_*\|^2\right] \leqslant 2e^{-\frac{T}{\kappa}}\|x_0 - x_*\|^2 + \tilde{\mathcal{O}}\left(\frac{L\tau_{\text{mix}}\left(\frac{1}{4}\right)\sigma_*^2}{\mu^3 T}\right).$$

### 2.4.3 Examples of communication graphs

So far in this section, we have assumed that the graph was given, and we built algorithms and convergence results that depend on it. In practice, the graph can be predefined if the learning task leverages an existing network. Examples include sensor networks, decentralized internet structures, popular social networks like Facebook or Twitter, or collaborations between institutes or hospitals. Sometimes, however, the graph can be chosen and constructed by the participants. In this case, it is interesting to choose the graph that is best suited to the problem at hand. For example, in the secure aggregation protocol of [Bel+20], participants reach out to $k$ of their neighbors, forming a $k$-out random graph, which turns out to be sparse but rather well-connected and robust to collusions. Similarly, it is interesting to optimize various properties of the graph in decentralized learning. In this section, we first introduce a few classical graphs and networks from real-world use cases, then some techniques to generate random graphs, and finally some properties of these networks.

Among the most classical graphs used in this thesis are trees, with $n - 1$ edges to connect all vertices. They can be as flat as a star or as high as a line (see Figure 2.5). An additional edge on the line gives the ring. Then, the torus makes graph connections more redundant. The hypercube [Yin+21], for a number of nodes that is a power of two, has a small number of edges but also a small radius – the radius is the minimum among all the maximum distances between a vertex to all other vertices. Finally, connecting all the nodes gives the complete graph.

Classical graphs can also stem not from their mathematical construction but from real networks. There are several real-world graph datasets, such as those available on [LK14].

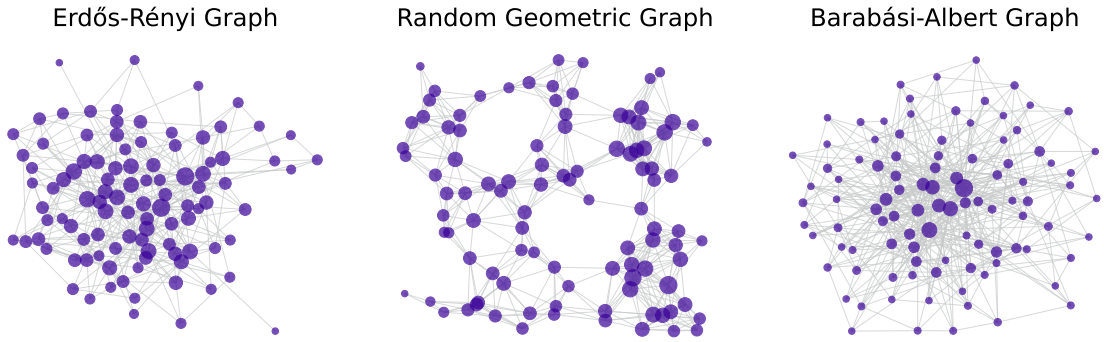



**Figure 2.6** – Example of random graphs with $n = 100$ nodes, size of node is proportional to its degree

These networks can represent social networks, citation networks, collaboration networks, road networks, etc. We will often test our approach on the Facebook Ego dataset from this repository, where nodes are friends of a given user (this central user is not present in the graph) and edges encode the friendship relations between these nodes. Ego graphs typically induce several clusters corresponding to distinct communities, such as the same high school, the same university, or the same hobbies.

In many cases, it is convenient to generate the graph randomly through a procedure that can be performed locally at the node level, using only a few parameters [BP16]. In particular, random graph generation is useful to make the graph vary through time and to study the dynamics of social networks by generating synthetic graphs that capture some of their properties, particularly asymptotic behaviors. Perhaps the simplest random graph is the Erdos-Rényi [ER59; HKP19], where each edge is added to the graph according to an independent Bernoulli random variable of probability $q$ (see Figure 2.6). These graphs are almost surely fully connected when $q > \log(n)/n$ and are *expanders* – expander graphs are sparse graphs that have strong connectivity properties – for well-chosen $q$ (e.g., $2\log(n)/n$). In this graph, nodes are somewhat all the same, as edges are created independently from each other: this is convenient in some scenarios that benefit from this symmetry, but it does not match behaviors usually seen in social networks, where already popular nodes tend to attract even more connections. Barabasi-Albert graphs capture this by adding each node sequentially and creating edges to a node proportionally to its current number of neighbors, which also gives a scale-free degree distribution. Finally, another way to generate random graphs is to inject randomness via the position of the node in space. Then, geometric random graphs connect all nodes below a given radius. There are a lot of other random graph generation algorithms, capturing various graph properties, that will not be used in this thesis.

When choosing the graph is possible, it is thus interesting to optimize for some properties. An important graph parameter that has been highlighted in Section 2.4.1 is the spectral gap: in

**Table 2.2** – Spectral gap (Definition 2.10) and degrees of classical graphs

| Graph | Complete | Expander | $D$-dim. Torus | Ring | Line |
|---|---|---|---|---|---|
| $\lambda_W$ | 1 | $\mathcal{O}(1)$ | $\mathcal{O}\left(\frac{1}{n^{\frac{2}{D}}}\right)$ | $\mathcal{O}\left(\frac{1}{n^2}\right)$ | $\mathcal{O}\left(\frac{1}{n^2}\right)$ |
| Degree | $n-1$ | $\mathcal{O}(1)$ | $2D$ | 2 | 2 |

order to converge quickly, one seeks a graph with a spectral gap close to $1$ and not $0$. To some extent, the degree can capture the communication cost. In Table 2.2, we recap the values for some graphs presented above. An expander graph is a sparse graph that has strong connectivity properties, which is achieved by the hypercube or by Erdos-Rényi graphs with a good parameter. It is thus an interesting trade-off in terms of communication cost and speed of convergence.

# Chapter 3

# Privacy

In this chapter, we discuss the concept of privacy, starting from a more general philosophical point of view before narrowing it down to the technical formulation of differential privacy and its use in machine learning. In the first part, we motivate the importance of privacy by offering examples and foundational definitions based on sociological works, concrete privacy breaches, and legal aspects (Section 3.1). We then move towards the definition of differential privacy, motivating its technical introduction, giving examples of mechanisms, and important properties relevant to the thesis (Section 3.2). Finally, focusing on machine learning, we explore privacy attacks and their connection to differential privacy, introduce differentially private stochastic gradient descent, and discuss the trust models used in machine learning research (Section 3.3).

## 3.1 Contextualizing Privacy

In this section, we adopt a non-technical point of view, to later motivate not only the introduction of differential privacy, but also the need to continue to play with its definition and to adapt it, as done in the thesis. It aims to clarify some high-level motivations for privacy-preserving machine learning. The goal is thus not to finish with a closed definition of privacy, but to give shape the broad concept of privacy by comparing it to other notions and confronting it with examples, and to provide good references to readers interested by the social impacts of machine learning in terms of privacy.

### 3.1.1 Elements of privacy in philosophy and sociology

> Privacy is valuable not only for our personal lives, but for our lives as citizens – our participation in public and community life. It is hard to imagine how people could freely participate in public life without some degree of control over their reputation and private life. Thus privacy is more than a psychological need or desire; it is a profound dimension of social structure. In addition to protecting individuals, privacy safeguards relationships between individuals, which are essential for family life, social engagement, and political activities. [Sol08]

Privacy is a fundamental *human right* and recognized as such in the Universal Declaration of Human Rights [NU48]. It should be enforced and respected, just like the right to have a nationality, access to fair trials, and freedom of speech, thought, and religion. As often, this right is intertwined with others, with each right helping the others to exist, but sometimes being orthogonal to each other and requiring a balance of different aspects [RD23].

From a personal point of view, my first tangible experience of privacy might come from "Les Misérables". Jean Valjean has a past, he stole some bread to feed his family and then spent years in jail, trying to escape several times and thus increasing his duty time. However, he also grows morally during the novel to represent the best traits of humanity, becoming known under the name of Monsieur Madeleine. He deserves to be allowed to keep his past private, but society is reluctant to let people change once they have been classified, especially as criminals. In Victor Hugo's society, the surveillance comes from the state, and is embodied by the character of Javert, who keeps recognizing him and trying to arrest him. When Javert finally realizes that Jean Valjean cannot be reduced to his past, he cannot deal with the contradiction between the Law that commands putting him in jail and his intimate conviction that Jean Valjean is a good man, and commits suicide.

Keeping information private is usually less dramatic than in Hugo's novels. Whether using sealed letters or encrypted communication, booths to vote, clothes to dissimulate the body, keeping the draft of a book private before publication, running away from paparazzi, or attending meetings only for members of an association, we use our right to privacy [Sol08] on a daily basis. The need for privacy is always dependent on the context [Nis09]: stars are not against photographers, but against photos of their intimacy; a book is meant to be read once published; people might choose to undress in specific situations.

The examples of book drafts or voting demonstrate that privacy cannot be reduced to the concept of *intimacy*. The set of elements that one could legitimately keep private is not limited to those directly related to personal intimacy. Another confusion is to reduce privacy to *secrecy*, which in particular gives the false idea that keeping information private is bound to immoral dissimulation. This confusion often arises when asking to end the secrecy of communication — for instance, by making cryptography illegal — by arguing that this would be beneficial in fighting against child pornography or terrorism. This is, for instance, the justification of

the Patriot Act [Con01], which led to *mass surveillance* in the United States, as described by Edward Snowden [Sno19]. How surveillance is eventually misused for other purposes such as fighting political opponents is well known [Ram24; BB19], and we refer the reader to [Zub19; Shu20]. Thus, I will only argue that mass surveillance is inefficient due to Bayes' law: there are not enough criminals in society for systematic checks to be efficient. Even with a very low probability of false positives, the rate of false positives when scanning the whole population with independent tests is overwhelming. Hence, making everything public by default is not an efficient way to fight awful but truly scarce behavior.

A mean to achieve privacy is *anonymization*. It is not always sufficient: if the goal was to keep a text private and it is leaked, even if the anonymity of the author is preserved, harm might still be done. However, when thinking of machine learning use cases, where privacy risks comes from a model that generalize from data, the privacy risk is often due to the ability to link specific data to an individual [WM19]. That is why we often talk about *unlinkability* as well for privacy-preserving mechanisms. For instance, the fact that some people have cancer does not need to be kept private; it is a well-known fact and useful to have data from to advocate for screening and advance research, but revealing the fact that a given individual has cancer does harm his/her privacy [DR14].

Another related concept is *obfuscation*. If secrecy and anonymization cannot be achieved, making access to the information harder might still be worthwhile. This represents a weaker form of privacy; it may eventually be compromised, but it is hoped to offer protection for a necessary short period. Examples include various military decoys during the Second World War and the Vula operation during Apartheid [BN15]. When proposing a digital service, relying on obfuscation for protection is however seen as bad practice: even if clients' data is not leaked today, it will be at some point. Hence, as we will show in Chapter 4, claiming that decentralized algorithms are private because it is hard to track the information leakage is not satisfactory. Accountability and trust are built on transparency, ensuring that proper protections measures are in place.

As discussed above, examples related to privacy are various, and indeed the situations in which this right is used have little in common. However, it always has the same goal, to create the necessary room for an individual to become who they choose to, a room for peace, trials and errors, dreams [Woo29]. It has been described as *the shield against the tyranny of the majority* by the American Supreme Court, and indeed it is by monitoring who accesses which information that a citizen constructs their place in a society [Zub19]. The diverse nature of privacy makes it impossible to confine it to a single concept such as intimacy, unlinkability, or anonymization as previously seen. Thus, a more effective approach is to focus on specific privacy problems [Sol08].

Helen Nissenbaum has proposed *contextual integrity* [BGN17] to accurately describe such problems, framing them as the appropriateness of information flow. Contextual integrity affirms that we need five elements to accurately determine whether a flow of information is appropriate:

1. *The sender of the data* corresponds to the entity that actually shares the information. It might have received it from another entity; it can be a legal entity, such as an institution or a company.

2. *The data subject*, on the contrary, is the individual that produced the data, for instance, the person in the picture that is shared.

3. *The recipient of the data* is the entity or individual who receives the data.

4. *The information type* corresponds to the content of the information, for instance, the picture.

5. *The transmission principle* corresponds mainly to the conditions under which the flows happen, such as reciprocity, consent, by law, or purchase.

From this formalism, one should remember that it is not possible to reduce the question of appropriateness to the information type, as it is often done when trying to legitimate inadequate use of data. Even if data is made to be public for some friends, it does not mean that we expect a company to train a large language model on it, or it is not because we share an illness in a discussion group that we consent to its use in computing an insurance rate [TKC24; VW19; Hao20].

Paying attention to the five elements allows us to understand why privacy has become hard to grasp. New *senders* are collecting and even trading data, for instance, to sell targeted advertisements. Machine learning offers new *transmission principles* that is not yet well understood, and so mysterious in its implications that most people cannot meaningfully consent [Bur16]. Even the *information type* now includes new kinds of data, such as precise geolocation over time [Mon+13], health data from smartwatches, web browsing logs, and extensive records of communication.

In the face of this deluge of data [Sch16], it has been said that privacy is dead [MAK20][1], and that even worse, younger generations do not feel concerned with this right as they happily share intimate parts of their lives on social networks [Boe05]. Even if their actions seem to translate an unwillingness to protect their privacy, they still have concerns about it. This *privacy paradox* translates the difficulty of protecting personal data today. People do not have the power to control the flow of their information, so they let it go [HM16].

The fact that a technological upheaval brings new needs to protect privacy is not new. In 1890, the American lawyers Warren and Brandeis wrote their famous article on *the right to be*

[1]We cite this work as it has found a wide audience, but disagree with both its conclusions and methods.

*left alone* in reaction to the popularization of cheap newspapers with photos, which led to the publication of articles about mundane facts.

> Instantaneous photographs and newspaper enterprise have invaded the sacred precincts of private and domestic life; and numerous mechanical devices threaten to make good the prediction that what is whispered in the closet shall be proclaimed from the house-top. [WB90]

Big data and machine learning are in many ways even more significant changes, which also require new protections. In the introduction, we already discussed the three upheavals brought by big data: first, the recent ability to collect data via a wide range of devices such as smartphones and IoT, and novel practices such as constant rating, systematic loyalty programs, and precise online tracking [Ful; Mon+13; Aim20; Swe00]; second, unprecedented data availability through cheap storage and nearly no latency to access data, unified formats, and larger infrastructures; and third, a wider ability to extract value from data with the development of machine learning algorithms. This allows to influence people [Lin24], to classify with the risk of discriminating [Zub19; Hao20], and to force people into invisible jails of their own recommended virtual worlds [Sim15; Par12]. This motivates the study of these emerging flows of information and the emergence of novel privacy problems.

### 3.1.2 Privacy in Law

> Code is law. [Les06]

Although the previous section, by highlighting the social impact of privacy, serves as a motivation to study privacy, the rise of privacy concerns from the industry's point of view is also mainly explained by legal requirements. The General Data Protection Regulation (GDPR) was signed in 2016 and enforced in 2018 in the European Union. It fostered research in privacy, by giving a clear motivation: finding a way to continue to extract value from data while being GDPR compliant.

The text has several interesting aspects. It starts with motivations close to the ones exposed in the previous section, but also clearly highlights the will to strike balance between economic interests and protecting privacy:

> Rapid technological developments and globalisation have brought new challenges for the protection of personal data. The scale of the collection and sharing of personal data has increased significantly. Technology allows both private companies and public authorities to make use of personal data on an unprecedented scale in order to pursue their activities. Natural persons increasingly make personal information available publicly and globally. Technology has transformed both the economy and social life,

> and should further facilitate the free flow of personal data within the Union and the transfer to third countries and international organisations, while ensuring a high level of the protection of personal data.

The legislator continues to rely heavily on pseudonymisation:

> 'Pseudonymisation' means the processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information is kept separately and is subject to technical and organisational measures to ensure that the personal data are not attributed to an identified or identifiable natural person.

It has been ten years since the AOL incident (see next section): it is already known that pseudonymisation has shortcomings, and it is thus a political choice to continue to give credit to this type of protection. However, the GDPR also defines *anonymous data* as data that "does not relate to an identified or identifiable natural person or to personal data rendered anonymous", which is closer to the unlinkability presented before, and illustrates that pseudonymisation is not the only approach. Another choice is to center the problem around personal data, and in particular on *Personally identifiable information* (PII), and consent rather than on privacy as a whole [Slo16]. This focus inspires a vision where data could be a property like any other, with the same exclusivity principle: the data belongs to someone, so it does not belong to someone else. This is not verified in practice; for instance, genetic data are shared among family members, and seeing data as a good raises many problems (see [NC15] and in particular Rochfeld's contribution). However, recognizing a single data owner allows us to hold them responsible for sharing the data, which brings us to the importance of consent in GDPR: data can be collected and processed (subject to extensive discussion of the situations) as long as the data subject *consents*. In practice, for non-technical users, GDPR is thus often synonymous with strange pop-ups on websites, that luckily close after clicking on "accept all". It is extremely hard to deny consent [HM16], or even to know what permissions have been given, far from the *freely given, specific, informed and unambiguous* agreement required by the text. GDPR fails to see that people have already lost control over the collected data, and that processing activities are so numerous that a single individual cannot trace all their personal data.

In practice, GDPR has given mainly two actionable means to resist privacy assaults. One is given by the *right of access by the data subject* that allows all users to download all their data, which is a way to know *a posteriori* to which data collection we have consented to. Properly analyzed, it has been often an important first step to understand the level of precision in the data collection.[2]

[2]From my generation, people downloaded their Facebook personal information, were terrified about it, and wanted to stop using Facebook. Unfortunately, no alternative scaled and we still are on Facebook.

The second one is to establish a supervisory authority in every country (Article 51) – in France, it is the CNIL who already existed thanks to the law "Internet et Libertés" [CNI15] – that is in charge to check compliance at the national level. The ability to fine big companies in case of violations has given a financial and a reputational incentive to not only respect GDPR but to be proactive and anticipate GDPR issues in advance.

Other data regulations also apply outside of the European Union, such as The California Privacy Rights Act of 2020 (CPRA), or the Personal Information Protection and Electronic Documents Act (PIPEDA, the first version became law in 2000). This thesis also terminates shortly before the introduction of the European AI Act, that focuses on the specific risks due to the application of artificial intelligence, forbidding most unethical applications such as social credit, emotion recognition on working places, predictive policing…It also creates new obligations for high risk applications, such as the ones using biometric data, or used for human resources. Again, the financial impact might be high, as the maximal fine is caped at 7% of global revenue of the firm.

Regulations cannot solve all privacy issues: illegal behavior happens, and fines do not suppress previous data leakages. Moreover, although legislation protects consumers from private companies, the dynamics are different for state surveillance. Often created in reaction to tragic terrorist attacks, many countries are keen to collect large amounts of personal data and to force private companies to hand over data when asked. The most well-known example is the Patriot Act [Con01], but similar dynamics are at play in other countries, especially in the Five Eyes[3], which tend to push towards more surveillance [Shu20; Zub19; Aim20]. In many ways, these five big democracies are more intrusive than many dictatorships, which brings us back to the privacy paradox: why do citizens let their representatives build the ultimate state of bureaucracy, as described in the Permanent Record [Sno19]?

One explanation, already hinted in [Sno19], might come from the fact that despite tracking a lot of illegal behaviors, there is no massive repression such as in non-democratic states, and surveillance stays invisible for the majority. When abortion rights were jeopardized by the overturn of Roe and Casey in 2022, the United States' possible repression against women showcased that current data collection could become a threat for users, making the data collection dangers more discernible. Some companies such as Google promised to delete this sensitive information, but it seems that the results did not match the expectation with incomplete and thus useless deletion [Gua24]. This is a strong incentive to keep user data decentralized and not accessible by the companies: even if the original purpose was genuine, a state subpoena can transform the data into a mean of persecution. In our example of a shopping basket, even if the prediction task is not pregnancy, it is obviously possible to detect it when a woman stops buying sanitary products.

[3]The Five Eyes (FVEY) is an anglosphere intelligence alliance comprising Australia, Canada, New Zealand, the United Kingdom, and the United States.

### 3.1.3 Data Leakage

> Data can be either useful or perfectly anonymous but never both. [Ohm09]

In this subsection, we move to more concrete problems raised by Big Data by discussing data leakage, informally corresponding to the unexpected disclosure of information about a specific individual. We have seen in the first subsection that a concept related to privacy in the context of data collection is anonymization and unlinkability. If one can remove the link between the data subject—i.e., the individual to whom the data refers—while keeping the data unaltered, we would realize the best of both worlds in the context of machine learning. Indeed, we would be able to learn our model with data, or to access it for statistical analysis or data visualization, but we would not be able to tell who participated in the dataset, thus effectively protecting participants. In this section, we show how this approach led to catastrophic failures in the past. Indeed, the possibility of anonymization relies on the illusion that one could separate an individual from their data. This is not the case [OP16].

Pseudonymization refers to the process of removing information that would allow a human being to directly link a record to an individual, as previously described in Section 3.1.2. This typically means removing names, addresses, social security numbers, credit card numbers, or similar entries. The term pseudonymization has been chosen in the community to raise awareness of the fact that this operation does not enable, most of the time, the severing of ties between the remaining entries and the individual. It is now well-known that in most cases, it is possible to link back a specific record to an individual, which is called reidentification or de-anonymization of the data. Examples of reidentification are numerous; we list three famous ones below:

**Governor of Massachusetts' health record** Latanya Sweeney demonstrated through analysis of the 1990 US Census that 87.1% of people in the United States could be uniquely identified by their ZIP code, birth date, and sex [Swe15]. When the governor of Massachusetts promoted the public release of health records by claiming it was risk-free due to pseudonymization, Latanya Sweeney crossed the database with the voters' list, found his record, and mailed it to him [Swe00].

**AOL database** In 2006, AOL, once an alternative to Google, released a pseudonymized dataset with users' search histories, where each user was designated by a random number. Some search histories contained harmful or embarrassing queries —e.g., "how to kill my wife" or "dog that urinates on everything". Despite initial defenses claiming it was impossible to identify who made the queries, two New York Times reporters managed to identify a retired woman, Thelma Arnold, who acknowledged that it was indeed her search history, illustrating that even retired women perform some embarrassing queries [BZ06].

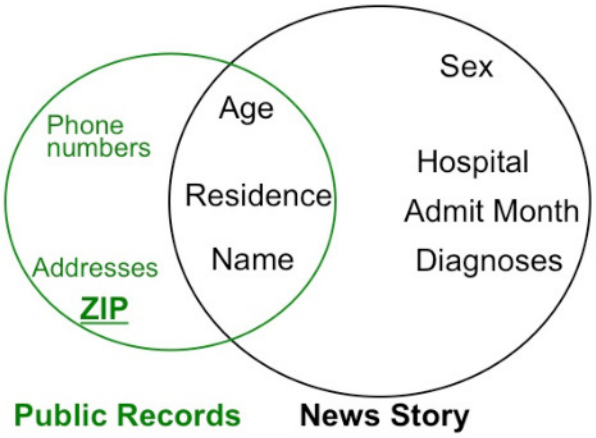



**Figure 3.1** – Join between public and private databases from [Swe15].

**Netflix dataset** A few months after AOL, Netflix launched a challenge to predict users' movie ratings from their past ratings. At the time, Netflix was still a DVD-by-mail company, but predicting the best movie to recommend was already crucial to business success, so a one million dollar prize was offered for a 10% accuracy increase in rating prediction accuracy. Researchers demonstrated that by cross-referencing the dataset with another public rating dataset, Internet Movie Database (IMDb), they could link IMDb public profiles to their complete movie histories in the Netflix database [NS06].

We will focus on the first example, because it emphasizes how women have pioneered research in privacy: after discussing the contributions of Helen Nissenbaum in the previous chapter, we see that Latanya Sweeney was not only the first African-American woman to obtain a PhD from MIT but also put forward defenses and analyses of privacy [Swe02b; Swe02a; SLP18; SAW13]. This example is particularly striking because the governor was a specific victim, not just a random unlucky person; the data was clearly sensitive as it was a medical record, and the reidentification was explicit.

How did the attack work? The released data still contained exact birth dates, zip codes, and genders, which seemed legitimate to keep the data valuable for researchers: knowing a person's age is clearly useful for health analysis. However, nearly 9 out of 10 people are uniquely identified by these quasi-identifiers. It is possible to join the public voters' database and the health records on the key built on these three features. (birth date, zip code, sex), as done in Section 3.1.3. This join puts on the same row the name contained in the voter list and the health information.

This example shows that even if some sets of features are not equivalent to a name for a human being, they are quasi-identifiers[4] that are unique and can play the same role. Faced

[4]The term quasi-identifier is sometimes used to refer to each of the feature rather than their union. Here we stick to the choice made in Sweeney's works where it refers to the whole set.

with this problem, a first verification that can be done prior to data release is to make sure that no record with easily accessible public features is unique. $K$-anonymity was introduced by Latanya Sweeney to formalize this idea.

**Definition 3.1** ($K$-anonymity [Swe02b])**.** *A dataset $\mathcal{D}$ is $K$-anonymous with respect to a quasi-identifier if each sequence of values of the quasi-identifiers appears at least $K$ times in $\mathcal{D}$.*

This approach effectively addresses the previous attack, but might require to modify significantly the database [Swe02a], and it relies on the strong to unrealistic assumption that one can identify the relevant quasi-identifier. Moreover, it fails to address a very simple case: if the $K-1$ records that share your quasi-identifier have the same disease as you, there is no protection. $L$-diversity fixes this by ensuring that the set has at least $L$ different records for each quasi-identifier [Mac+07], and there exist other refinements [LLV07]. However, too many constraints are hard to meet and can lead to the deletion of all the rows in the database.

These methods highlight two persistent problems: quasi-identifiers are ubiquitous, and trying to make records indistinguishable destroys the utility of the dataset. Differential privacy proposes another method to manage this tradeoff.

## 3.2 Differential Privacy

### 3.2.1 Definition

> "Differential privacy" describes a promise, made by a data holder, or curator, to a data subject: "You will not be affected, adversely or otherwise, by allowing your data to be used in any study or analysis, no matter what other studies, data sets, or information sources, are available." [DR14]

$K$-anonymity is not sufficient to efficiently protect personal information. Rather than trying to enforce more complex and arbitrary constraints like $L$-diversity does, Differential Privacy creates uncertainty in all the rows by injecting noise into queries[5]. The initial idea of using randomness to protect privacy is usually traced back to the art of surveys [War65]. In 1965, Warner proposed a new method to avoid bias caused by respondents changing their answers because they prefer not to disclose shameful or illegal behavior to the interviewers. To minimize this behavior, which is hard to estimate and could invalidate the significance of the results, respondents are asked to modify their answers based on the external randomness of a coin flip, as described in Algorithm 3.1. The goal is to estimate the true proportion of positive answers and not the individual answers.

[5] In this context, "queries" refer to data requests or analyses made on a dataset

**Algorithm 3.1:** Randomized response as originally proposed by [War65]

```
Toss two coins sequentially
if first result is tail then
    Tell the truth
else
    if second result is tail then
        Say yes
    else
        Say no
```

This algorithm creates plausible deniability: if your answer is the shameful one, you can deny its truthfulness by claiming it was given by the random coin toss. This algorithm creates additional variance to the estimator because half of the answers are pure noise, but it should remove the bias term caused by lying participants in the process, since answering 'yes' does not create significant data leakage for the person. The original mechanism relies on a fair coin, but we could consider using a biased coin for the initial toss: the more it is biased towards revealing the truth, the less private it is, allowing more information to be inferred about the respondent's true answer.

Differential privacy builds on this idea and generalizes the answer to yes-no questions to any kind of output, for instance the ones given by machine learning algorithms, whether they are predictions or weights. To do so, it relies on the notion of adjacent datasets, which we discuss before moving to the definition of differential privacy. Two datasets are said to be adjacent if they identical except for changes in one record, which reflects the granularity we want to protect [DR14]. In the motivational example of randomized response, the private information is a single bit, unlike the complex datasets typically involved in machine learning.

Most of the time, it is safe to think about adjacent datasets as representing two scenarios in which the only difference is the modification of one of the rows of the dataset of the first scenario into another one. However, how granularity is translated mathematically depends on the information you want to protect: if two datasets are adjacent when they differ from a single shopping basket, we aim at protecting specific choices made on a given day, and one talks about *record-level* differential privacy. One could also decide to consider *user-level* privacy [McM+18], by considering that all the shopping baskets of a single user should be seen as a single entity rather than a single basket. In these case, we move from the first scenario to the second one by replacing one of the participant by another one. At an even more ambitious level, *group-level* privacy [DR14] considers the appropriate granularity to be the entire family, comprising several closely related users who share mutual information about each other. In this case, two datasets are adjacents if all rows corresponding to a given family can be modified between while other rows remain identical.

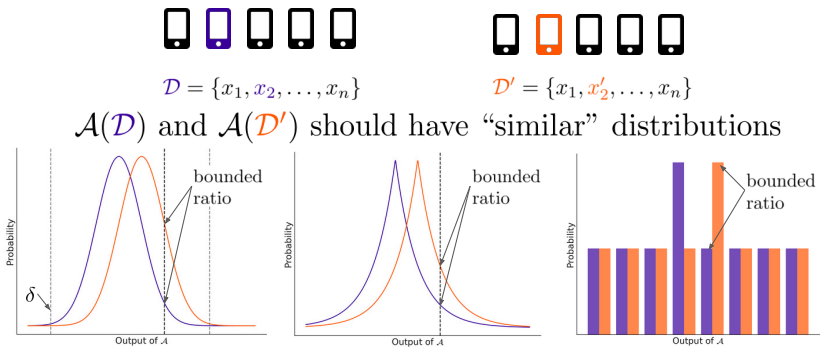



**Figure 3.2** – Illustration of differential privacy intuition: between the two scenarios, the datasets stay identical except for the blue user that is replaced by the orange one. The output remains with a bounded ratio, as illustrated with classically used mechanisms in differential privacy—Gaussian, Laplace, and Randomized Response from left to right—where for every output the blue and the orange values remain not too far apart.

Differential privacy then imposes constraints on the outputs of the algorithm between these two scenarios. We note $\mathcal{D} \sim \mathcal{D}'$ to indicate that the datasets $\mathcal{D}$ and $\mathcal{D}'$ are adjacent.

**Definition 3.2** (Differential Privacy [Dwo+06a])**.** *Let $\varepsilon \geq 0$ and $\delta \in [0, 1]$. An algorithm $\mathcal{A}$ is $(\varepsilon, \delta)$-differentially private with respect to $\sim$ if for every pair of databases $\mathcal{D} \sim \mathcal{D}'$ and all $\mathcal{S} \subset \text{Range}(\mathcal{A})$:*

$$\mathbb{P}(\mathcal{A}(\mathcal{D}) \in \mathcal{S}) \leq \exp(\varepsilon)\mathbb{P}(\mathcal{A}(\mathcal{D}') \in \mathcal{S}) + \delta$$

The variable $\varepsilon$ is the *privacy budget*, that quantifies in one positive number the privacy leakage that occurs by revealing the outputs of the algorithm. It is called the budget as it is a quantitative measure that one needs to manage, for instance by spending more budget on some learning tasks rather that others, and that it can be used as a guideline to ensure that we do not overexpose some data. Translating this number into a level of risk with a clear intuition is still a rather open problem. Consensus is often to say that the acceptable privacy budget should depend on the context, but even when fixing a rather precise context, experts opinions might vary a lot. Some claims that even a big privacy budget is better than an infinite one. However, another consensus is that it was fallacious for Apple to claim respecting privacy when using a budget of 16 per day [Tan+17]. In this thesis, we keep $\varepsilon$ as a variable and most

of the time we do not require specific assumptions on this side, leaving the task of determining the appropriate value to practitioners.

Differential Privacy in Definition 3.2 presents an additive term $\delta$. When setting it to $0$, we have pure differential privacy, which is more restrictive: $\delta > 0$ offers the freedom to have some catastrophic failures, as long as their probability to happen remain extremely small in practice.

Comparing the two outputs for the two databases boils down to the tricky problem of comparing two distributions. Even it remains implicit in the historical formulation of differential privacy presented above, it is more natural, and often more convenient to think in terms of divergence.

**Definition 3.3** (Hockey-stick divergence)**.** *The hockey-stick divergence with $a \geq 1$ between two distributions $\mu$ and $\nu$ is defined by*

$$D_a^h(\mu \| \nu) = \sup_E (\mu(E) - a\nu(E)) = \int \max(\mu(y) - a\nu(y), 0)\, dy$$

With this definition, an algorithm is differentially if and only if for every pair of adjacent databases $\mathcal{D} \sim \mathcal{D}'$, $D_{e^\varepsilon}^h(\mathcal{A}(\mathcal{D}) \| \mathcal{A}(\mathcal{D}')) \leq \delta$.

Other divergences can be used to derive differential privacy guarantees. A popular choice is the Rényi divergence.

**Definition 3.4** (Rényi Differential Privacy (RDP))**.** *An algorithm $\mathcal{A}$ satisfies $(\alpha, \varepsilon)$-Rényi Differential Privacy (RDP) for $\alpha > 1$ and $\varepsilon > 0$ if for all pairs of neighboring datasets $\mathcal{D} \sim D'$:*

$$D_\alpha(\mathcal{A}(\mathcal{D}) \| \mathcal{A}(\mathcal{D}')) \leq \varepsilon$$

*where $D_\alpha(\mu \| \nu)$ is the Rényi divergence:*

$$\mathcal{D}_\alpha(\mu \| \nu) = \frac{1}{\alpha - 1} \log \int \left(\frac{\mu(z)}{\nu(z)}\right)^\alpha \nu(z) dz$$

Interestingly enough, DP and RDP are closely related, as one can translate results from RDP to $(\varepsilon, \delta)$ DP with the following lemma.

**Lemma 3.5** (Conversion RDP to DP)**.** *If an algorithm is $(\alpha, \varepsilon)$ Rényi differentially private, it is $\left(\varepsilon + \frac{\log 1/\delta}{\alpha - 1}, \delta\right)$-differentially private for any $0 < \delta < 1$.*

Choosing one definition over another is primarily a matter of selecting the best tool for a given task, as each differential privacy setting offers interesting properties for analyzing

algorithms, such as difference in composition [MTZ19]. Other definitions, relying on other ways to compare distributions, also exist, such as $f$-differential privacy, but will not be investigated in this thesis.

### 3.2.2 Two examples of differentially private mechanisms

We already introduced the Randomized Response (RR) mechanism for binary queries. More generally, for any query on a finite set, we can return the true answer with some probability and a random choice uniformly chosen among the other options the rest of the time.

**Proposition 3.6** (RR is DP)**.** *Let $RR$ be the algorithm over $\{1, \dots, K\}$ such that for an input $x$, the outputs are drawn with the following probability:*

$$RR(y|x) = \begin{cases} \frac{e^\varepsilon}{K-1+e^\varepsilon} & \text{if } y = x \\ \frac{1}{K-1+e^\varepsilon} & \text{if } y \neq x \end{cases}$$

*This algorithm is $\varepsilon$-differentially private.*

Similarly, for continuous outputs, we will rely on Gaussian noise, which is easy to analyze and to generate.

**Proposition 3.7** (Gaussian Mechanism)**.** *Let $f$ be a function with $l_2$-sensitivity of $\Delta$, defined by*

$$\Delta = \max_{\mathcal{D} \sim \mathcal{D}'} \|f(\mathcal{D}) - f(\mathcal{D}')\|$$

*The Gaussian mechanism, given by $\mathcal{A}(D) = f(D) + \mathcal{N}\left(0, \Delta^2\sigma^2\right)$ is $(\varepsilon, \delta)$-differentially private for $\delta > 0, \varepsilon < 1$ and $\sigma \geq c/\varepsilon$ with $c^2 > 2\log(1.25/\delta)$. It is also $(\alpha, \alpha/2\sigma^2)$-Rényi differentially private.*

In practice, we use these mechanisms to design differentially private algorithms by making each of the interactions with the data private. In machine learning, where we need to interact more than once with each data point, we use these methods multiple times, thus requiring composition results. This is one of the many nice properties of differential privacy.

### 3.2.3 Properties of Differential Privacy

Differential Privacy does not distinguish between sensitive and non-sensitive information, nor does it differentiate between Personally Identifiable Information (PII) and non-PII data. The examples discussed in Section 3.1.3 demonstrate that this is part of its strength, as it

is impossible to predict in advance which features could lead to data leakage during post-processing. According to the contextual integrity framework (see Section 3.1), one could say differential privacy is a means of transmission that remains agnostic to the content of the message. When introducing differential privacy in [DR14], Dwork and Roth provide an example that fits our discussion of the shopping basket dataset.

> Revealing "ordinary" facts, such as purchasing bread, may be problematic if a data subject is followed over time. For example, consider Mr. T, who regularly buys bread, year after year, until suddenly switching to rarely buying bread. An analyst might conclude Mr. T most likely has been diagnosed with Type 2 diabetes. The analyst might be correct, or might be incorrect; either way Mr. T is harmed.

We also saw that previous data anonymization approaches could turn out to be inefficient because their obfuscation techniques proved to be unsuccessful once the correct attack was applied. Differential Privacy suppresses that risk, because the theoretical guarantees provided actually correspond to the removal of information: the process cannot be reversed, as indicated by the post-processing property.

**Proposition 3.8** (Post-Processing)**.** *If $\mathcal{A}$ is a differentially private algorithm and $f$ a function independent from the dataset $\mathcal{D}$ and from the randomness of $\mathcal{A}$, then the composition $f \circ \mathcal{A}$ is also differentially private with the same parameters.*

This property ensures immunity against all attacks, even those still unknown today, making it a highly desirable feature.

**Theorem 3.9.** *Let $f : \mathcal{X} \to \mathcal{Y}_1$ and $g : \mathcal{X} \times \mathcal{Y}_1 \to \mathcal{Y}_2$ be respectively $(\alpha, \varepsilon_1)$ and $(\alpha, \varepsilon_2)$-Rényi differentially private. Then, the release of $g(f(\mathcal{D}), \mathcal{D})$ is $(\alpha, \varepsilon_1 + \varepsilon_2)$-Rényi differentially private.*

It is thus sufficient to sum up all the $\varepsilon$'s in RDP, which makes this variant of DP quite convenient. Summing $\varepsilon$'s is also correct for DP, but not tight. Note in particular that the computation of the second mechanism can depend on the output of the first mechanism.

Composing all the interactions of a realistic algorithm can lead to large privacy budget. To improve it, we can also leverage amplification mechanisms which hide certain intermediate computations in order to make an algorithm more private. A first example of such mechanism is subsampling, where we apply our algorithm only to a subset of the data that is randomly sampled from the total dataset.

**Theorem 3.10** (Privacy Amplification by Subsampling [BBG18; FMT20])**.** *Let $q \in (0, 1/5)$ be the sampling rate, $\sigma \geq 4$ and $\mathcal{S}$ the mechanism where each element is sampled independently at*

*random with probability $q$ without replacement and $f$ a function of sensitivity $\Delta$. Then $f \circ \mathcal{S}(\mathcal{D}) + \mathcal{N}(0, \Delta\sigma^2)$ is $(\alpha, \frac{6\alpha q^2}{\sigma^2})$-RDP.*

This amplification relies on the secrecy of the sample. There exists numerous other theorems on subsampling, that cover the spectrum of possible hypotheses, some of them requiring numerical procedures, but the takeaway is always that the amplification is roughly proportional to $q^2$ in RDP.

Another interesting mechanism to protect privacy is to keep the intermediary steps of a computation hidden and only reveal the last step. In this case, despite not being directly revealed, the previous steps can still be partially leaked through the final output, but there is some privacy amplification, that is captured by privacy amplification by iteration.

**Theorem 3.11** (Privacy Amplification by Iteration [Fel+18; AT22])**.** *Let $T^1, \dots, T^K, T'^1, \dots, T'^K$ be 1-Lipschitz operators, an initial random state $x^0 \in U$ , and $(\zeta^k)_{k=1}^K$ a sequence of noise distributions. Consider the noisy iterations $x^{k+1} = T^{k+1}(x^k) + \eta^{k+1}$ and $\bar{x}^{k+1} = T^{k+1}(\bar{x}^k) + \bar{\eta}^{k+1}$ where $\eta^{k+1}$ and $\bar{\eta}^{k+1}$ are drawn independently from distribution $\zeta^{k+1}$. Let $s_k = \sup_{x \in U} \left\| T^k(x) - \bar{T}^k(x) \right\|$. Let $(a_k)_{k=1}^K$ be a sequence of real numbers such that*

$$\forall k \leq K, \sum_{i \leq k} s_i \geq \sum_{i \leq k} a_i, \text{ and } \sum_{i \leq K} s_i = \sum_{i \leq K} a_i \,. \tag{3.1}$$

*Then,*

$$D_\alpha(x^K || \bar{x}^K) \leq \sum_{k=1}^{K} \sup_{x : \|x\| \leq a_k} D_\alpha(\zeta_k * \mathbf{x} \| \zeta_k) \,, \tag{3.2}$$

*where $*$ is the convolution of probability distributions and $\mathbf{x}$ denotes the distribution of the random variable that is always equal to $x$.*

Privacy amplification by iteration (PABI) is a way to decrease distance in the distribution space as time increases: when each iteration is 1-Lipschitz, [6] the difference between the updates done on different data of a given step fades progressively at each iteration and this phenomenon can be quantified by analyzing the process with a *shifted* Rényi divergence, that computes a divergence between a distribution and a distribution moved towards the other by a bounded Wasserstein distance [Fel+18]. The amplification is roughly linear in the number of iterations done before revealing the result of the computation. While this mechanism was introduced in the context of DP-SGD (see Section 3.3.2), in this thesis we show that this amplification mechanism has also applications for decentralized algorithms (see Chapter 6 and Chapter 7) and for other iterative algorithms such as ADMM (see Chapter 8).

[6] A 1-Lipschitz operator is often referred as a non-expansive operator.

### 3.2.4 Local Differential Privacy

The constraints of differential privacy apply to the outputs of the algorithm $\mathcal{A}$. When introducing amplification mechanisms in the previous section, we saw that hiding intermediary steps of the algorithms allows us to decrease the privacy leakage of these steps. We call Central Differential Privacy the setting where only the final outputs of the algorithm need to be protected. Conversely, in a decentralized setting, one might consider that all messages sent by a participant could be intercepted, and thus the guarantees should apply to all messages sent by a given node.

Local Differential Privacy [Kas+08; DJW13] relies on similar mechanisms than central DP, such as randomized response, Laplace or Gaussian injection, but the scale of the noise is thus computed over all possible entries of a space $\mathcal{X}$ rather than on two adjacent datasets.

**Definition 3.12** (Local Differential Privacy [Kas+08; DJW13])**.** *Let $\varepsilon > 0$ and $\delta \in (0, 1)$. A local randomizer algorithm $\mathcal{R}$ is $(\varepsilon, \delta)$-locally differentially private (LDP) if for all $x, x' \in \mathcal{X}$ and any possible $S \subset \mathcal{Z}$:*

$$\mathbb{P}(\mathcal{R}(x) \in S) \leq e^{\varepsilon}\mathbb{P}\left(\mathcal{R}\left(x'\right) \in S\right) + \delta$$

Local differential privacy can thus be seen as the equivalent of differential privacy for a database of size 1 and is convenient for decentralized algorithms [HMV15; Bel+18]. This assumption corresponds to the setting where we do not trust the central aggregator, and thus each participant protects his or her own data. This, however, comes at a price in terms of utility. Here, we use utility as a generic term to design how efficient the algorithm is, which can be for instance captured by the mean squared error of an estimator, or the accuracy of a model in machine learning. To illustrate the gap in utility between local and central DP, consider the task of computing privately the average $\bar{x}$ over $n$ samples in $[0, \Delta]$ held locally by participants. Under $(\alpha, \varepsilon)$-RDP constraint, the best possible utility is bounded by:

$$\mathbb{E}(\left\|x^{\text{out}} - \bar{x}\right\|^2) \leq \frac{\alpha\Delta^2}{2n\varepsilon} \quad \text{for local DP}\,, \quad \text{and} \quad \mathbb{E}(\left\|x^{\text{out}} - \bar{x}\right\|^2) \leq \frac{\alpha\Delta^2}{2n^2\varepsilon} \quad \text{for central DP}\,,$$

where $x^{\text{out}}$ is the output of the algorithm. This $1/n$ gap motivates the study of intermediate trust models, where LDP is relaxed so as to obtain better utility while still avoiding the need for a trusted curator. A popular approach is to resort to cryptographic primitives to securely aggregate user contributions [Dwo+06b; Shi+11; Bon+17; CSS12b; Jay+18; SBR22] or to securely shuffle the set of user messages so as to hide their source [Che+19b; Erl+19; Bal+19b; Bal+19a; Gha+20; FMT20] (Figure 3.3). Recent work has also considered the so-called shuffle model of DP [Che+19b; Erl+19; Bal+19b; Bal+19a; Gha+20; FMT20], where users send their contribution to a trusted curator or secure shuffler which permutes the set of messages so as to hide their source. At the cost of additional computation or communication overhead,

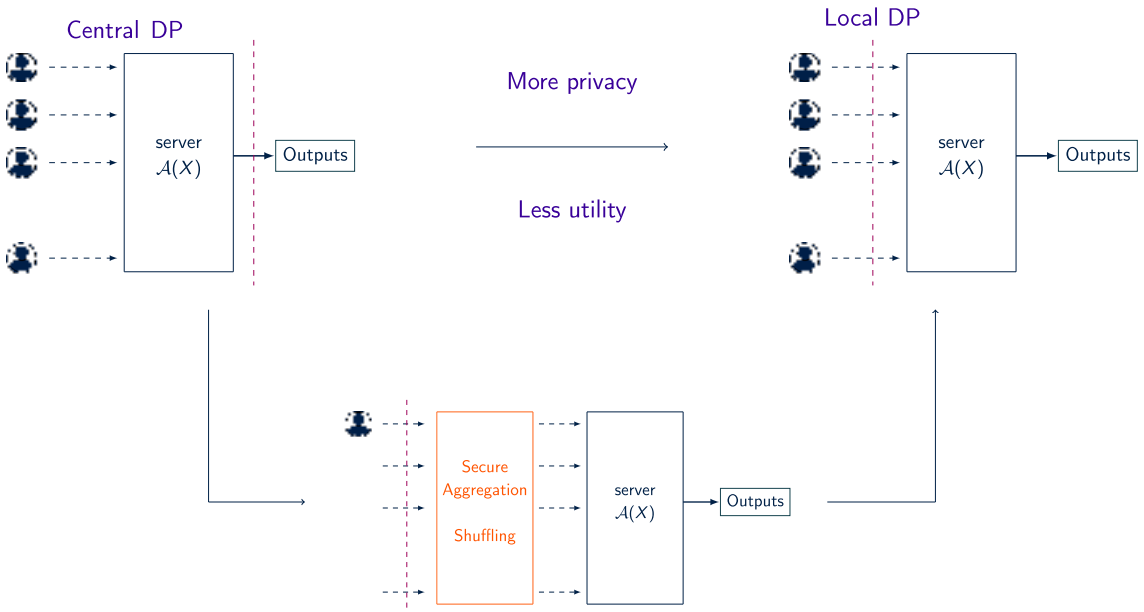



**Figure 3.3** – Intermediate trust model between local and central differential privacy.

these relaxations can provably lead to significant improvements in the privacy-utility trade-off (sometimes matching the trusted curator model).

These existing trust models, however, rely on the presence of a central aggregator to perform secure aggregation or on the possibility of adding a trusted third party to perform the shuffling. This is not feasible in fully decentralized settings with only peer-to-peer communication (as defined in Section 2.4). Developing intermediate trust models well adapted to this setting is one of the main contributions of this thesis.

## 3.3 Privacy in Machine Learning

### 3.3.1 Privacy Attacks in Machine Learning

Our examples of privacy issues have been focused so far on data leakages, where data is extracted through a simple join between databases, for example. One of the vulnerabilities of machine learning stems from the fact that, despite its apparent complexity, it is often possible to extract information about individual training data points from a trained network, or from its updates [Sho+17]. In this section, we provide a quick overview of existing attacks[7]. If a successful attack can leak significant information from the dataset, then the algorithm cannot

[7]Note that these privacy attacks should not be confused with robustness attacks, where a malicious agent tries to evade detection or misclassify an object, such as an image of a panda being classified as a gibbon. These are unrelated to the attacks we discuss here

be considered private. This complements the study of differential privacy previously seen in Section 3.2.

We can distinguish various types of privacy attacks, that always aim to extract information about the training data points, from the most impressive to the easiest to achieve:

**Data Reconstruction** [MV14; ZLH19; ZMB20] where the attacker aims to retrieve about individual training points only from the model. This attack is particularly visual in the case of image reconstruction and thus has gained popularity. It is also the most complete one, as other attacks are also successful if reconstruction is complete.

**Attribute inference attack** [Mel+19; KÁF23] only targets some properties of the data points, for instance, the label, or a protected attribute.

**Membership inference** [Sho+17; Car+21; Ngu+23; ZLS24] aims only at determining if a given data point is part of the training dataset or not; it can be seen as the least ambitious attack. However, it is the attack with the closest connection to differential privacy. Indeed, the privacy budget $\varepsilon$ captures how much some outputs have different probabilities whether a data point is part of the training set. In practice, however, the difficulty in deriving the tightest $\varepsilon$ value means that often results in observed attack performance are less impressive than what the privacy budget implies. Attacks thus come as an empirical way to bridge the gap between the theoretical guarantee of differential privacy and the evidence given by the attack.

The goal of an attack is however not the only factor to take into account. What the attacker can access and do for attacking greatly changes the quantity of information that can be extracted. An attacker is said to be passive, or honest-but-curious, when the attacker does not modify the behavior of the training and only learns information from what is already available. On the contrary, an active attacker could, for instance, send false messages (for example false gradients in Algorithm 2.1) to obtain more information [Boe+21]. In this section, we will focus on passive attacks, as it notably avoids discussing to what extent malicious updates can be performed without being detected, and matches the setting studied in Chapter 4.

Another factor that greatly changes the ability to attack is how much the attacker knows about the model. It is possible to have non-trivial results in attacks even when considering a black box model, where the attacker only accesses predictions of the final model. It is however limited, and often requires training *shadow models* that mimic the true model based on predictions to learn about the training process. However, it is easier to attack when the model and training procedure is known. This is a motivation to study the federated setting: if the attacker is a participant or the server, they have access to the structure and the parameters (see Algorithm 2.1). In particular, the gradient received by the server offers a new surface for attack compared to the centralized case.

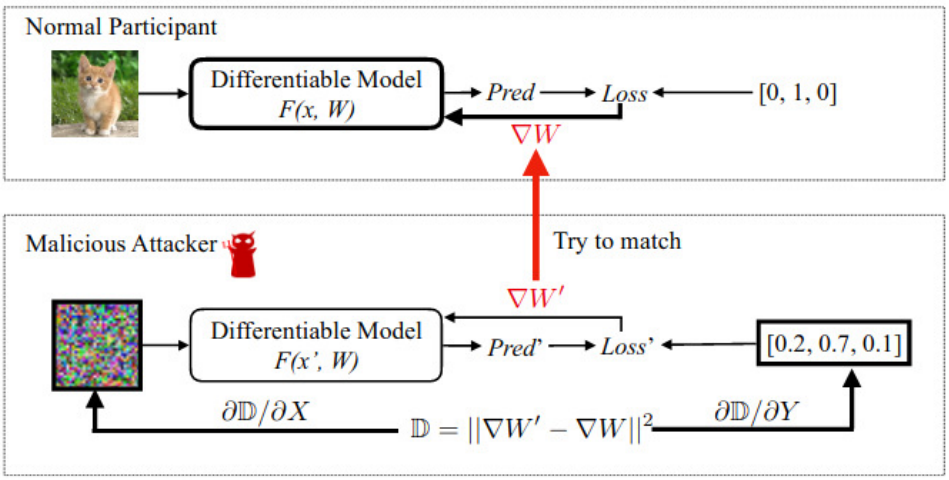



**Figure 3.4** – Schema from [ZLH19]. The reconstruction attack starts from the noisy picture to converge to the cat one by minimizing the difference between the two gradients $\nabla W$ and $\nabla W'$.

It is easy to see that if the function is an immersion, having access to the gradient allows determining which data point was used, by using the inverse of the derivative. This is the case in logistic regression or a full convolutional layer with at least one output that is activated. To see this, assume we want to reconstruct a data point $x$ and we have access to the update of the weights in $f(w^\top x + b) = y$. Assume that we can easily invert $f$, for example, if it is the sigmoid or an activated ReLU, then we have $w^\top x + b = y'$. The gradient we observe corresponds to

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}w} = \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}y'} \cdot \frac{\mathrm{d}y'}{\mathrm{d}w}$$

We can replace the first factor by the derivative with respect to the bias, which we also observe, noting that $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}y'} = \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}b}$. The second factor corresponds exactly to the desired $x^\top$, achieving the reconstruction. This method was introduced in [Gei+20b]. In fact, even when closed form reconstruction is impossible or too difficult to achieve, numerous attacks still allow successful reconstructions from the gradients. One way to proceed is to perform a gradient descent to invert the gradient: starting from a random data point, and computing its gradient, we modify the image to minimize the difference between the fake gradient and the true one (see Figure 3.4).

These methods are an active topic of research. Various improvements for gradient inversion have been proposed [Gei+20a; ZMB20; Wan+19], including techniques capable of separating gradients aggregated over large batches [Kar+23]. This motivates the need to adapt machine learning techniques to minimize these risks. We will turn to differential privacy for doing so.

### 3.3.2 Differentially Private Stochastic Gradient Descent (DP-SGD)

We now turn to the famous DP-SGD algorithm [BST14; SCS13; RA12] and give the main takeaways in terms of privacy-utility tradeoffs. We aim to solve empirical risk minimization as seen in Chapter 2 that we recap here:

$$\hat{\theta} \in \arg\min_{\theta \in \Theta} \widehat{\mathcal{R}}\left(f_\theta\right) = \frac{1}{n} \sum_{i=1}^{n} l\left(y_i, f_\theta\left(x_i\right)\right) \tag{3.3}$$

**Algorithm 3.2:** Differentially private SGD

**Input:** Data points $\{x_1, \ldots, x_N\}$, loss function $f$, learning rate $\gamma_t$, noise scale $\sigma$, batch size $B$, gradient norm bound $C$.
Initialize $\theta_0$ randomly
**for** $t \in [T]$ **do**
  Take a random sample $\mathcal{B}_t$ with sampling probability $B/N$
  **for** *each* $i \in \mathcal{B}_t$ **do**
    Compute $g_t(x_i) \leftarrow \nabla_\theta f(\theta_t, x_i)$ ▷ `Compute gradient`
    $\bar{g}_t(x_i) \leftarrow g_t(x_i) / \max(1, \frac{\|g_t(x_i)\|_2}{C})$ ▷ `Clip gradient`
  $\bar{g}_t \leftarrow \frac{1}{B}\left(\sum_i \bar{g}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbb{I})\right)$ ▷ `Add noise`
  $\theta_{t+1} \leftarrow \theta_t - \gamma_t \bar{g}_t$ ▷ `Descent`
**Output:** $\theta_T$

The private version of SGD is quite close to the non-private version, and mainly present two differences: the clipping of individual gradients and the Gaussian noise addition to the gradient. Clipping can be avoided if a theoretical bound is known on the gradient magnitude (e.g., when the loss function is $L$-Lipschitz), but if it often more efficient in practice to just aggressively clip the gradient. This introduces bias and a parameter to tune, and it is unclear how we should change this clipping bound through the iterations [McM+18; TAM19]. Noise addition introduces more variance in the updates, and the accumulated noise causes the iterates to converge within a ball whose radius is proportional to the noise level. Unlike non-private stochastic gradient descent, this noise cannot be reduced by using smaller step sizes and more iterations, as each step increases the privacy budget due to composition. Note that the fact that we cannot converge to an exact minimizer is not surprising: doing so would violate differential privacy. Finally, because the noise is without structure, it is sometimes needed to add a projection on a convex set. The privacy-utility trade-off can thus be summarized for a given level of noise as follows.

**Theorem 3.13** (Utility and Privacy of DP-SGD [BST14]). *Let* $\sigma^2 = O\left(\frac{L^2 n^2 \log(n/\delta)\log(1/\delta)}{\varepsilon^2}\right)$. *For* $\theta_T$ *output by Algorithm 3.2, for a loss function* $\mathcal{L}$ *that is* $L$*-Lipschitz, we have:*

$$\mathbb{E}\left[\mathcal{L}\left(\theta_T; \mathcal{D}\right) - \mathcal{L}\left(\theta^*; \mathcal{D}\right)\right] = O\left(\frac{L \log^{3/2}(n/\delta)\sqrt{d\log(1/\delta)}}{n\varepsilon}\right)$$

*and if* $\mathcal{L}$ *is also* $\mu$*-strongly convex, we have*

$$\mathbb{E}\left[\mathcal{L}\left(\theta_T; \mathcal{D}\right) - \mathcal{L}\left(\theta^*; \mathcal{D}\right)\right] = O\left(\frac{L^2 \log^2(n/\delta) d \log(1/\delta)}{n^2 \mu \varepsilon^2}\right)$$

*Furthermore, the output is* $\left(\alpha, \frac{\alpha T}{2n^2\sigma^2}\right)$*-Rényi differentially private.*

The privacy analysis relies the Gaussian mechanism, the composition property of RDP, and amplification by subsampling. By using only a subset of size $B/n$ of the data points, we have an amplification of $B^2/n^2$ but the sensitivity of each update is also scaled by the sampling factor, so both cancel out in the final guarantee given in the above theorem.

This theorem reveals the trade-off between privacy and utility: both formulas depend on $\sigma$; as $\sigma$ increases, privacy improves but utility worsens, showcasing the tension between these two objectives. It can be demonstrated that the above bounds are optimal up to some logarithmic factors [BST14]. However, it is generally difficult to determine the best achievable trade-off and how to approach it, depending on the chosen trust assumptions. For instance, it might be possible to achieve some level of privacy for free in the over-parameterized regime [BM24]. Identifying which algorithms and conditions provide the best trade-offs is a key question in privacy-preserving learning.

In this thesis, we will study these questions in the decentralized setting, thus introducing various private versions of existing decentralized algorithms (Section 2.4) and trying to obtain privacy-utility guarantees that are of the same order as those of centralized algorithms.

# Chapter 4

# Privacy Attacks in Decentralized Learning

In this chapter, we demonstrate that decentralization alone is not enough to prevent data leakage. We study Decentralized Gradient Descent (D-GD), an algorithm that allows a set of users to perform collaborative learning without sharing their data. D-GD proceeds by iteratively averaging local model updates with their neighbors in a network graph, as seen in Section 2.4. We propose the first attack against D-GD that enables a user (or set of users) to reconstruct the private data of other users outside their immediate neighborhood. Our approach is based on a reconstruction attack against the gossip averaging protocol, which we then extend to handle the additional challenges raised by D-GD. We validate the effectiveness of our attack on real graphs and datasets, showing that the number of users compromised by a single or a handful of attackers is often surprisingly large. We empirically investigate some of the factors that affect the performance of the attack, namely the graph topology, the number of attackers, and their position in the graph. Our code is available at https://github.com/AbdellahElmrini/decAttack.

These results motivate the need of adding differential privacy to decentralized algorithms, as done in subsequent chapters. This chapter is mostly based on [MCB24].

## 4.1 Introduction

We saw in the previous chapter the importance to protect data in machine learning, and that federated learning was susceptible to reconstruction attacks (Section 3.3.1). In this chapter, we investigate whether gossip-based algorithms, by preventing any single node from observing the individual contributions of all other nodes, are less vulnerable than traditional federated learning systems. Indeed, while a node can observe the contributions of its immediate neighbors in the graph, making the reconstruction of their values somewhat expected [PRT23], the

contributions of more distant nodes are iteratively mixed multiple times with other contributions before being observed. Intuitively, one might thus expect that individual contributions are harder to retrieve from only indirect feedback, especially when the attacker node is far from its target. This, in turn, leads to the common belief that decentralized learning may be more privacy-preserving by design. Questioning this belief is the motivation of our work.

To do so, we design and evaluate attacks performed by a node (or a set of nodes) to retrieve other nodes' private data in two algorithms: gossip averaging, a key protocol in decentralized computation [Boy+06; XB04; OT09], and the prominent Decentralized Gradient Descent (D-GD) algorithm [Lia+17; Kol+20]. As our goal is to assess the potential vulnerabilities specific to decentralization, we focus on the strongest type of privacy leakage, namely data reconstruction, executed by the weakest type of attackers, namely honest-but-curious nodes. In other words, attacker nodes follow the protocol and try to reconstruct as much data as possible from other nodes using only their legitimate observations within the protocol. To the best of our knowledge, the attacks we present are the first to be able to reconstruct data from non-neighboring nodes.

Our attacks rely on interpreting each message received by the attackers as an equation that ties together the private values of nodes, the weights of the gossip matrix, and the received values. In the case of gossip averaging, we show that an appropriate factorization of a knowledge matrix representing these equations yield the private values of the (often numerous) reconstructible nodes, and additionally produces a reduced set of equations linking together the values of the remaining nodes (which may leak sensitive information in certain cases). Interestingly, we can predict even before running the algorithm which nodes will have their values leaked depending on the gossip matrix of the network graph. For the case of D-GD, the key ideas of our attack against gossip averaging can be applied to reconstruct individual gradients, with some modifications to account for the extra difficulty induced by combining gossip averaging with gradient descent steps. Once we have reconstructed individual gradients, we rely on existing gradient inversion attacks [ZLH19; Gei+20a] as a black-box to reconstruct data points. We show that under reasonable assumptions, the reconstruction of the private data points of non-neighboring nodes is still possible in D-GD.

We empirically evaluate the performance of our attacks on synthetic and real graphs and datasets. Our attacks prove to be effective in practice across various graph structures. In many cases, even a single attacker node is able to reconstruct the data of a large number of other nodes, some of them located many hops away. The collusion of several attacking nodes further strengthens the attack. We also observe that the graph topology, the position of attackers in the graph, and the choice of learning rate play an important role.

Our results clearly show that relying solely on decentralization to ensure the privacy of training data is ineffective, even if nodes fully trust their immediate neighbors. Thus, additional protections must be implemented.

Concurrently and independently from our research, Dekker, Erkin, and Conti [DEC23] studied a different setting where nodes perform a sequence of secure aggregations with their neighbors, without considering a learning objective. Similar to [PRT23], their attack focuses solely on direct neighbors. In contrast, our attack is capable of reconstructing the data of more distant nodes, thus addressing the specific challenges of the decentralized setting more effectively.

**Related work.** We are aware of only two recent works on privacy attacks targeting decentralized learning [PRT23] and Dekker, Erkin, and Conti [DEC23]. The attack of [PRT23] only targets direct neighbors, making it quite similar to existing attacks on federated learning. Concurrently and independently from our research, Dekker, Erkin, and Conti [DEC23] studied a different setting where nodes perform a sequence of secure aggregations with their neighbors, without considering a learning objective. Similar to [PRT23], their attack focuses solely on direct neighbors. In contrast, our attack is capable of reconstructing the data of more distant nodes, thus addressing the specific challenges of the decentralized setting more effectively.

## 4.2 Setting

### 4.2.1 Decentralized Algorithms

As introduced in Chapter 2, we consider a fixed undirected and connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $|\mathcal{V}| = n$ and an edge $\{u, v\} \in \mathcal{E}$ indicates that $u$ and $v$ can exchange messages. We denote by $\mathcal{N}(u) = \{v : \{u, v\} \in \mathcal{E}\}$ the neighbors of node $v$. We assume that each node $u$ possesses a private value $x_u \in \mathbb{R}^d$. To ease the notations, we will assume in the presentation of the next sections that $d = 1$, but the generalization to the vector case is straightforward.

**Remark 4.1.** *The simplification of considering local datasets with a single element was also made by Pasquini, Raynal, and Troncoso [PRT23] and is natural in the case of decentralized averaging, where each node has indeed only one private value. For Decentralized Gradient Descent, recent work has proposed reconstruction attacks from gradients aggregated over several data points [Boe+21; Kar+23], which can be readily used as a black-box in our attack. We thus abstract away this secondary aspect to focus on the particularity of decentralized learning.*

We consider synchronous gossip-based algorithms, where at each step each node performs a weighted average of its value with those of its neighbors, as determined by the weights given by a gossip matrix (see Definition 2.9).

**Gossip averaging.** The gossip matrix can be used to iteratively compute the average of the private values $(x_v)_{v\in\mathcal{V}}$. Initializing each node $v$ with $\theta_v^0 = x_v$, the gossip averaging iteration [XB04] is given by

$$\theta^{t+1} = W\theta^t. \tag{4.1}$$

This converges to the global average at a geometric rate governed by the spectral gap of $W$ (see [XB04] and Section 2.4).

**Remark 4.2** (Accelerated gossip)**.** *An accelerated version of gossip* [BBG20]*, which involves replacing the multiplication by $W$ with a polynomial of $W$, is often used to speed up convergence. We note that this does not impact the information accessible to a node, as the values observed in accelerated gossip can be easily translated back to the non-accelerated counterpart through a simple linear transformation. For the sake of clarity, our discussion will thus focus on the non-accelerated version.*

**Decentralized Gradient Descent (D-GD).** In decentralized learning, nodes aim to optimize an objective function of the form $f(\theta) = \sum_{v=1}^n L(\theta, x_v)$ where $\theta$ represents the parameters of the model and $L$ is some differentiable loss function. The most popular algorithm to achieve this is D-GD [Lia+17; Kol+20]. At each iteration of D-GD, each node performs a local gradient step with a learning rate $\eta > 0$ followed by a gossip averaging step with its neighbors. Let $\theta_v^0$ be an arbitrary initialization of the parameters at each node $v$. We denote the local gradient of node $v$ at iteration $t$ (scaled by $\eta$) by

$$g_v^t = -\eta\nabla_{\theta_v^t} L(\theta_v^t, x_v). \tag{4.2}$$

Then, the gradient step of D-GD can be written as

$$\theta^{t+\frac{1}{2}} = \theta^t + g^t, \tag{4.3}$$

and the gossiping step corresponds to

$$\theta^{t+1} = W\theta^{t+\frac{1}{2}}. \tag{4.4}$$

Under suitable assumptions, D-GD converges to a global or local optimum of $f$ [Kol+20].

### 4.2.2 Threat Model

Throughout this chapter, we focus on honest-but-curious attackers that consist of a subset of nodes that adhere to the protocol but attempt to extract as much information as they can from their observations. We denote by $\mathcal{A} \subset \mathcal{V}$ the set of attacker nodes and by $\mathcal{N}(\mathcal{A}) = \bigcup_{a\in\mathcal{A}} \mathcal{N}(a)\setminus\mathcal{A}$

the neighbors of the attackers. Without loss of generality, we assume that attacker nodes correspond to the first $|\mathcal{A}|$ nodes. When $|\mathcal{A}| > 1$, we assume that the knowledge is completely shared between attacker nodes, even if the network graph does not contain edges between them.

We also assume that the network graph and the gossip matrix are known to the attacker. This assumption is often naturally satisfied in practical use cases (e.g., within social networks), and we further discuss its relevance in Appendix A.4.

## 4.3 Reconstruction in Gossip Averaging

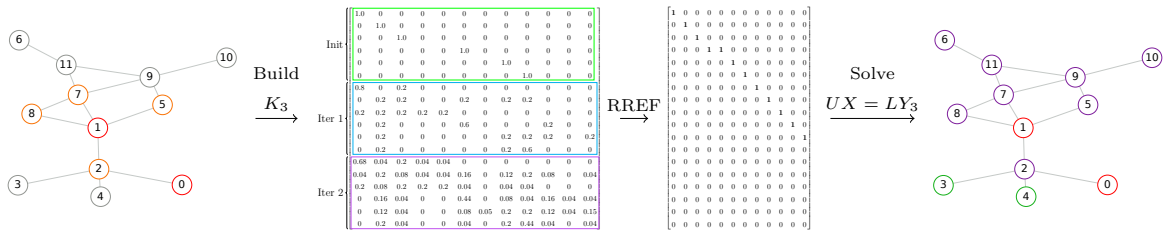



**Figure 4.1** – Overview of our attack on gossip averaging. The attackers 0 and 1 (red) receive updates from nodes 2, 5, 7 and 8 (orange). For $T = 3$ iterations, it leads to the knowledge matrix $K_3$. Its RREF (matrix $U$) exhibits that only nodes 3 and 4 are non-reconstructible (green). All other nodes (purple) have their private value leaked.

In this section, we describe our attack on gossip averaging. The key idea of our attack is that each message $\theta_v^t$ received by an attacker node $a \in \mathcal{A}$ from one of its neighbors $v \in \mathcal{N}(a)$ corresponds to a linear equation where the unknown are the private values of the nodes of the graph and the coefficients depend on the gossip matrix $W$ (which is known to the attackers). Our attack consists in accurately generating this system of linear equations ($K_T X = Y_T$ in our notations) and then solving it in order to reconstruct as many private values as possible. We provide a visual summary of this reconstruction process in Figure 4.1.

**Collecting linear equations.** For a given gossip matrix $W$ and set of attackers $\mathcal{A}$, we now describe how the attackers can systematically construct a system of linear equations capturing the knowledge that they gather about the private values throughout $T$ iterations of gossip averaging. Informally, the attackers receive $|\mathcal{A}| + T \cdot |\mathcal{N}(\mathcal{A})|$ values, and they can associate each of these values to a linear combination of the $n$ private inputs. At the beginning of the protocol, the attackers already know their inputs (these are the first $|\mathcal{A}|$ values). Then, at each round $t$ of gossip, the attackers receive one value $\theta_v^{(t)}$ from each of their neighbors $v \in \mathcal{N}(\mathcal{A})$. These received values correspond to a linear combination of private inputs, where the weights are given by the corresponding powers of the gossip matrix.

Formally, we denote this system of linear equations by $K_T X = Y_T$, where $X = (x_0, \dots, x_n) \in \mathbb{R}^n$ is the vector of private values that the attackers seek to reconstruct, and $K_T, Y_T$ are defined as follows.

**Definition 4.3** (Knowledge matrix and observation vector)**.** *The* knowledge matrix $K_T \in \mathbb{R}^{|\mathcal{A}|+T\cdot|\mathcal{N}(\mathcal{A})| \times n}$ *is defined by Algorithm 4.1. The* observation vector $Y_T \in \mathbb{R}^{|\mathcal{A}|+T\cdot|\mathcal{N}(\mathcal{A})|}$ *is obtained by having attackers stack their own values and the received messages* $(\theta_v^t)_{0\leq t\leq T-1, v\in\mathcal{N}(\mathcal{A})}$*. The pair* $(K_T, Y_T)$ *forms the view of the attackers.*

Note that $K_T$ only depends on the gossip matrix, whereas $Y_T$ is specific to the private values.

For concreteness, let us consider a simple example. Consider a graph of $n$ nodes with one attacker at position $0$. The attacker knows his own private input $x_0$ (stored in the first entry of $Y_T$), which corresponds to the insertion of the one-hot vector $(1, 0, \dots, 0)$ in the first row of the knowledge matrix $K_T$. Assuming that nodes $1$ and $2$ are the neighbors of the attacker, they will send $\theta_1^{(0)} = x_1$ and $\theta_2^{(0)} = x_2$ to the attacker at iteration $0$ (stored in the second and third entries of $Y_T$), corresponding to the insertion of $(0, 1, 0, \dots, 0)$ and $(0, 0, 1, 0, \dots, 0)$ respectively in the second and third rows of $K_T$. At iteration $1$, they will send $\theta_1^{(1)} = \sum_j W_{j,1} x_j$ and $\theta_2^{(1)} = \sum_j W_{j,2} x_j$, corresponding to $(W_{0,1}, \dots, W_{n-1,1})$ and $(W_{0,2}, \dots, W_{n-1,2})$ respectively. In general, for each iteration $t$, the attacker receives the values $\theta_1^{(t)} = \sum_j W_{j,1}^t x_j$ and $\theta_2^{(t)} = \sum_j (W^t)_{j,2} x_j$ from his two neighbors, by definition of the gossip averaging algorithm, and this information is stored in the corresponding rows of $K_T$ and $Y_T$.

**Solving the linear system.** Recovering the private values corresponds to solving the equation $K_T X = Y_T$ where $X$ is the unknown. The set of solutions of this equation is always non-empty by construction (as the set of private values satisfies the equation), and is reduced to a single element when $K_T$ is full rank (i.e., rank $n$). Thus, if $K_T$ is full rank, attackers can reconstruct all the private values of the nodes. Otherwise, attackers may still reconstruct a subset of the private values, and deduce relationships between the ones that cannot be fully reconstructed.

To do so, we factorize $K_T$ using its Reduced Row Echelon Form (RREF), namely $K_T = L^{-1}U$ where $U$ is the unique RREF of $K_T$ and $L$ is such that $UX = LY_T$. RREF is a form often introduced in algebra courses to teach Gauss-Jordan elimination [ND88]. In this form, the block of matrix $U$ corresponding to reconstructible nodes becomes the identity matrix, while the remaining rows (corresponding to non-reconstructible nodes) contain the linear equations that link their values together. Solving $K_T X = Y_T$ is thus equivalent to solving $UX = LY_T$ with trivial equations $1 \times X_v = (LY_T)_v$ for reconstructible nodes. Hence, this decomposition allows us to clearly identify the nodes that our attack can reconstruct, even before the algorithm is executed (as the attackers only need to construct $K_T$ but not $Y_T$).

**Algorithm 4.1:** Knowledge matrix construction

**Input:** the graph $\mathcal{G}$, the set of attackers $\mathcal{A}$, the number of iterations $T$
**Output:** The knowledge matrix $K_T$
$K_T \leftarrow$ an empty matrix of size $m \times n$ where $m = |\mathcal{A}| + T \cdot |\mathcal{N}(\mathcal{A})|$
**foreach** $v \in \mathcal{A}$ **do**
  $K_T[v, :] \leftarrow e_v$, where $e_v$ is the one-hot vector of size $n$ with 1 at index $v$
$i \leftarrow |\mathcal{A}|$
**for** $t = 0$ **to** $T - 1$ **do**
  **foreach** $v \in \mathcal{N}(\mathcal{A})$ **do**
    $K_T[i, :] \leftarrow W^t[v, :]$ ▷ `Gossip, at` $t = 0$, `attackers receive` $e_v$ `for` $v \in N(A)$ $i \leftarrow i + 1$
**return** $K_T$

**Definition 4.4** (Reconstructible node)**.** *Given a network graph $G$ and a set of attackers $A$, a node $v$ is said to be reconstructible by $\mathcal{A}$ after $T$ iterations if in the RREF form $U$ of $K_T$, the row corresponding to $v$ is a vector with the value 1 in a single entry and 0 everywhere else.*

A complete characterization of reconstructible nodes using explicit graph-related quantities, instead of linear algebra as in Definition 4.4, appears to be quite challenging. In Appendix A.1, we provide some counter-intuitive examples on simple graphs to illustrate this point.

**Remark 4.5** (Secure aggregation)**.** *One of the defense mechanisms widely studied in federated learning is the use of secure aggregation (SecAgg), where a set of parties jointly compute the sum of their private values without revealing more than the final output [Bon+17]. This could be used in gossip averaging: at each iteration, each node could compute the weighted sum by running a SecAgg protocol with its neighbors [DEC23]. Aside from the high computation and communication overhead, in terms of leakage, this would be akin to an attack in a model without SecAgg from a node only connected to the true attacker. We note that our attack still works in this case, with a slight modification in the construction of the knowledge matrix. As illustrated by Figure 4.4(b) or Figure 4.3, numerous nodes can have their data leaked in the case where the attacker has a single neighbor.*

**Remark 4.6** (Extension to dynamic networks)**.** *Our approach can be adapted to dynamic networks where the graph changes over time, as long as the attackers know the gossip matrix $W_t$ used at each iteration. Indeed, the reconstruction problem is equivalent to the one with a static $W$, up to the modification of the construction of the knowledge matrix. In some cases, dynamic networks might be more vulnerable to reconstruction: the union of the direct neighbors of the attackers could*

*be larger than for static networks, so the proportion of nodes that can be directly reconstructed might increase.*

## 4.4 Reconstruction in D-GD

In this section, we present our attack on Decentralized Gradient Descent. Our attack proceeds in two steps: we first reconstruct the gradients and then reconstruct the data points from the reconstructed gradients. The latter step can be done by resorting to existing gradient inversion attacks as a black box [see e.g., ZLH19; Gei+20a; Kar+23]. Therefore, in the rest of the section, we focus on the gradient reconstruction part, which is the core of our contribution.

To reconstruct the gradients of nodes, our attack builds upon the attack on gossip averaging presented in Section 4.3. However, several additional challenges arise. While the private values in gossip averaging remain constant across iterations, the gradients in D-GD evolve over time. This means that each step brings new unknowns in the equation, making it impossible to find an exact solution through direct equation solving. Attacking D-GD thus requires additional steps:

1. Reducing the number of unknowns by introducing similarity assumptions on the gradients;
2. Changing the construction of the knowledge matrix, to reconstruct the gradients $g_v^t$ instead of the model parameters $\theta^t$;
3. Removing the attackers' own contributions to reduce overall noise in the approximated reconstruction;

**1. Gradient similarity.** We derive our reconstruction attack under the assumption that the gradients of a node along the iterations can be described as a combination of a fixed and a random component.

**Assumption 4.7** (Noise-signal gradient decomposition)**.** *For each node $v \in \mathcal{V}$, we assume that we can decompose the gradient update as follows*

$$g_v^t = -\eta \nabla_{\theta_v^t} L(\theta_v^t, x_v) = g_v + N_v^t, \tag{4.5}$$

*where $N_v^t$ is a centered random variable with variance $\sigma^2$, and the constant part $g_v$ is specific to node $v$ but stays the same across iterations.*

We note that this assumption is typically not satisfied in real use cases, but we will see in Section 4.5 that the algorithm is robust in practice to small violations of this assumption (in particular, it works well when gradients change sufficiently slowly across iterations).

**Algorithm 4.2:** Building the knowledge matrix for D-GD

**Input:** the graph $\mathcal{G}$, the set of attackers $\mathcal{A}$, the set of targets $\mathcal{T} = \mathcal{V} \setminus \mathcal{A}$, the number of iterations $T$
**Output:** The knowledge matrix $K_T$
$K_T \leftarrow$ an empty matrix of size $m \times n$ where $m = T \cdot |\mathcal{N}(\mathcal{A})|$
$i \leftarrow 0$
**for** $t = 0$ **to** $T - 1$ **do**
  **foreach** $v \in \mathcal{N}(\mathcal{A})$ **do**
    $K_T[i, :] \leftarrow (\sum_{j=0}^{t} W_{\mathcal{T},\mathcal{T}}^{j})[v - |\mathcal{A}|, :]$ ▷ `Gossip, at` $t = 0$`, attackers receive` $e_v$ `for` $v \in N(A)$ $i \leftarrow i + 1$
**return** $K_T$

**Remark 4.8** (Connection to differential privacy)**.** *Assumption 4.7 naturally models situations where noise is added to satisfy differential privacy [HMV15; XZW21; Cyf+22]. In particular, one can see this as an instance of averaging under local differential privacy. Naturally, the accuracy of the reconstruction will be directly related to the variance of the noise, and thus to the chosen privacy budget.*

**2. Knowledge matrix construction.** Denoting the set of target nodes as $\mathcal{T} = \mathcal{V} \setminus \mathcal{A}$, we rewrite the gossip matrix $W$ as follows:

$$W = \begin{pmatrix} W_{\mathcal{A},\mathcal{A}} & W_{\mathcal{A},\mathcal{T}} \\ W_{\mathcal{T},\mathcal{A}} & W_{\mathcal{T},\mathcal{T}} \end{pmatrix} \tag{4.6}$$

We now write the values $\theta^{t+\frac{1}{2}}$ shared by nodes during the execution of the algorithm in terms of this decomposition.

**Proposition 4.9** (Closed-form of D-GD updates)**.** *For the D-GD algorithm described by (4.3) and (4.4), we have:*

$$\theta^{t+\frac{1}{2}} = \begin{pmatrix} \theta_{\mathcal{A}}^{t+\frac{1}{2}} \\ \left(\sum_{i=0}^{t} W_{\mathcal{T},\mathcal{T}}^{i}\right) g_{\mathcal{T}} + \sum_{i=0}^{t} W_{\mathcal{T},\mathcal{T}}^{i} N_T^{t-i} \\ + \sum_{i=0}^{t-1} W_{\mathcal{T},\mathcal{T}}^{t-1-i} W_{\mathcal{T},\mathcal{A}} \theta_{\mathcal{A}}^{i+\frac{1}{2}} \end{pmatrix}. \tag{4.7}$$

*Proof.* The proof is done by induction, applying the Equation (4.3) and Equation (4.4) and rearranging the terms. ■

This formula leads to a more complex computation of the knowledge matrix $K_T$ (see Algorithm 4.2) compared to the one used for gossip averaging, because we want to reconstruct the gradients $g = (g_v)_{v \in \mathcal{V}}$ (not model parameters $\theta^t$).

**3. Attackers' contributions removal.** Given $Y_T$ the concatenated vector of updates received by the attackers until iteration $T$, the attackers need to preprocess it in order to remove their own contributions. Algorithm 4.3 shows how to compute $\hat{Y}_T$ from $Y_T$ and the gossip matrix.

**Algorithm 4.3:** Removing the attackers' contributions

**Input:** the gossip matrix $W$ of the graph $\mathcal{G}$, the set of attackers $\mathcal{A}$, the set of targets $\mathcal{T} = \mathcal{V} \setminus \mathcal{A}$, the number of iterations $T$, the dimension of the model $d$, the received updates $Y_T$ and the concatenated vector of the updates sent by the attackers $\theta_{\mathcal{A}} = (\theta_{\mathcal{A}}^{\frac{1}{2}}, \dots, \theta_{\mathcal{A}}^{T-\frac{1}{2}})$

**Output:** The updated matrix $\hat{Y}_T$

$\hat{Y}_T \leftarrow$ an empty matrix in $\mathbb{R}^{T \times |\mathcal{N}(\mathcal{A})| \times d}$

$B \leftarrow$ zero matrix in $\mathbb{R}^{|\mathcal{T}| \times d}$

**for** $t = 0$ **to** $T - 1$ **do**

  $\hat{Y}_T[t, :] \leftarrow Y_T[t, :] - B[\mathcal{N}(\mathcal{A}), :]$

  $B \leftarrow W_{\mathcal{T},\mathcal{T}} B + W_{\mathcal{T},\mathcal{A}} \theta_{\mathcal{A}}^{t+\frac{1}{2}}$ ▷ `The contribution of the attackers to be eliminated`

**return** $\hat{Y}_T$

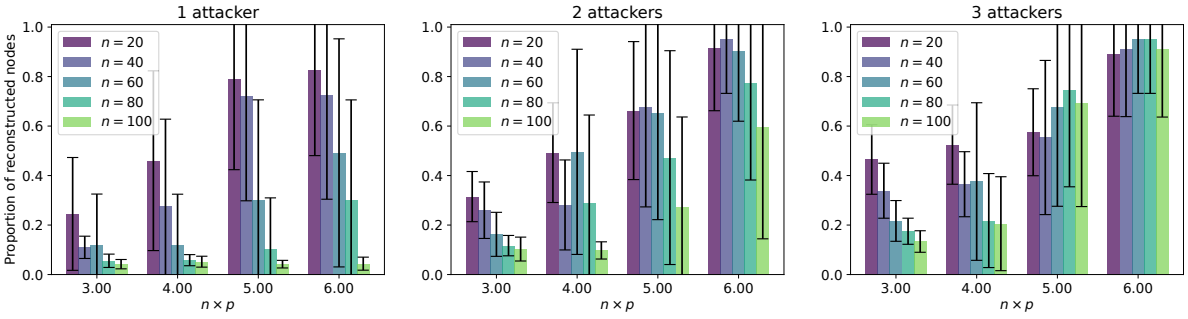



**Figure 4.2** – Average fraction of reconstructed nodes in Erdös-Rényi graphs with a different number of nodes $n$ and edge probability $p$, for $1, 2$ or $3$ attacker nodes. Error bars give the standard deviations, computed over $20$ random graphs.

**Gradient reconstruction.** Equipped with the previous concepts, reconstructing the gradients reduces to a Generalized Least Square (GLS) problem:

$$K_T g + \varepsilon_T = \hat{Y}_T, \tag{4.8}$$

where $\varepsilon_T$ is the noise of covariance $\Sigma_T$, the (non-diagonal) covariance matrix resulting from the aggregation of noise from various nodes at each step. The formula and detailed algorithm for computing this covariance matrix is provided in Appendix A.2 (Algorithm A.1).

The reconstruction minimizing the squared error is thus given by:

$$\hat{g} = (K_T^\top \Sigma_T^{-1} K_T)^{-1} K_T^\top \Sigma_T^{-1} \hat{Y}_T.$$

and this estimator is unbiased with a variance of size $(K_T^\top \Sigma_T^{-1} K_T)^{-1}$ under Assumption 4.7.

**Remark 4.10** (Impact of covariance matrix)**.** *An alternative to computing this exact covariance matrix $\Sigma_T$ is to solve directly the Ordinary Least Square (OLS). Although this gives a bit more weight to the most noisy points compared to the optimal estimator under Assumption 4.7, we found experimentally that the reconstruction quality is quite similar between the two methods. This can be explained by the fact that the assumption of the noise structure is not fully satisfied, and OLS tends to be quite robust in practice.*

**Remark 4.11** (Gossip protocols with consecutive averaging steps)**.** *An existing variant of D-GD involves performing multiple gossip averaging steps for each gradient step [Kon+21; Kol+20; Cyf+22; Dan+22]. Our attack can be easily adapted to this case. In fact, it makes reconstruction easier as it brings D-GD closer to gossip averaging by effectively making gradients constant across several iterations. In scenarios where sufficiently many iterations are performed, we can simply use the reconstruction attack designed for gossip averaging, up to the minor modifications to the knowledge matrix.*

## 4.5 Experimental Results

In this section, we show that our attacks on gossip averaging and decentralized gradient descent are effective in practice. We evaluate our attacks on synthetic and real-world graphs, showing successful reconstructions in all cases.

Our code is available at https://github.com/AbdellahElmrini/decAttack.

### 4.5.1 Gossip Averaging

**Synthetic graphs and impact of global characteristics.** We generate Erdös-Rényi graphs with different number of nodes $n$ and different edge probability $p$. We also vary the number of attacker nodes from 1 to 3. The proportion of reconstructed nodes in each setting is shown in Figure 4.2. We can see that a single attacker node is typically able to reconstruct many nodes,

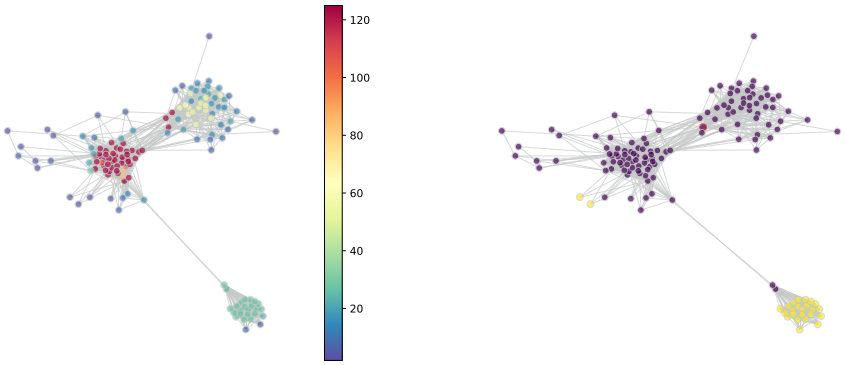



**Figure 4.3** – Reconstruction attack on the Facebook Ego Graph 414. Left: each node is colored by the number of nodes it can reconstruct among the 147 other nodes. Right: detailed view of the case where the node circled in red is the attacker, with reconstructed nodes shown in purple and non-reconstructed ones in yellow.

well beyond its direct neighborhood. The fraction of reconstructed nodes increases with the connectivity of the graph and the number of attackers.

**Real-world graphs.** We consider the graphs of the Facebook Ego dataset [LM12], where nodes are the friends of a given user (this central user is not present in the graph) and edges encode the friendship relation between these nodes. These graphs typically present several communities corresponding to different interests. We show that reconstruction is much more likely within clusters, but also occur across nodes belonging to distinct clusters. We give an example in Figure 4.3 and report other Ego graphs in Appendix A.6.

**Table 4.1** – Correlation between the centrality of the attacker and the proportion of nodes it is able to reconstruct.

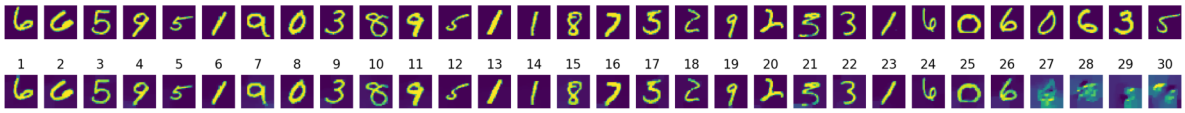
| Centrality | Erdos-Rényi graph | Ego graph |
|---|---|---|
| Degree | 0.94 | 0.94 |
| Eigenvector | 0.81 | 0.64 |
| Betweenness | 0.84 | 0.66 |

**Impact of nodes' characteristics.** Intuitively, it is easier to attack close nodes rather than distant ones. We quantify this effect by evaluating how the attacker centrality impacts its reconstruction ability. Centrality measures are often used in graph mining to measure how important a node is. We test two kinds of graphs. First, we randomly sample Erdös-Rényi graphs with $n = 50$ and $p = 0.08$. We reject graphs that are not fully connected and consider a single attacker (node 0 as the graph construction treats all nodes equally). Second, for a fixed Facebook Ego graph, we make each node play the role of the attacker in turn. To assess the relation between the centrality of a node and the proportion of the nodes it is able to

reconstruct, we use Spearman correlation, a non-parametric measure which is only based on rank statistics. We observe in Table 4.1 that for both types of graphs, degree centrality is the most correlated with the proportion of reconstructed nodes. Interestingly, other centrality measures that capture some structural properties of the graph beyond immediate neighbors, such as eigenvector and betweenness centrality, also exhibit strong correlation. In Appendix A.3, we report metrics to assess how the relative position of the attacker and its target impact the probability of reconstruction.

### 4.5.2 Decentralized Gradient Descent


(a) Cifar10, logistic regression, learning rate $10^{-4}$




(b) MNIST, convnet, learning rate $10^{-6}$, gradient inversion from [Gei+20a]

**Figure 4.4** – Reconstruction attack on D-GD for a line graph with 31 nodes where the attacker lies at an extremity. The first (resp. second) row shows the true (resp. reconstructed) inputs of the 30 other nodes ordered by their distance to the attacker.

We now turn to the more challenging case of D-GD. We first focus on the Cifar10 dataset [Kri09] using a model that consists of a fully connected layer with a softmax activation, a bias term, and a cross-entropy loss (a.k.a., logistic regression). For this simple model, one can reconstruct a data point from its gradient in closed-form [see Bis23, Lemma 6.1 therein]. This allows us to focus the evaluation on the core of our attack (reconstructing gradients), avoiding the inherent errors due to the imperfections of gradient inversion attacks on more complex models. We start running our attack when the model is close to convergence so that gradients are more stable. To ensure that attackers gather enough information about other nodes in the knowledge matrix, we run D-GD for a number of steps roughly equal to the diameter of the graph.

We first run our attack on the classic Florentine graph [BP86], a graph with $n = 15$ nodes describing marital relations between families in 15th-century Florence. We can see in Figure 4.5 that most nodes (except those located at the edge of the network) can reconstruct a large proportion of other nodes with very good visual accuracy.

We further test the limit of reconstruction by using a line graph of 31 nodes with the attacker at an extremity. We see in Figure 4.4(a) that the results are far better than one would intuitively

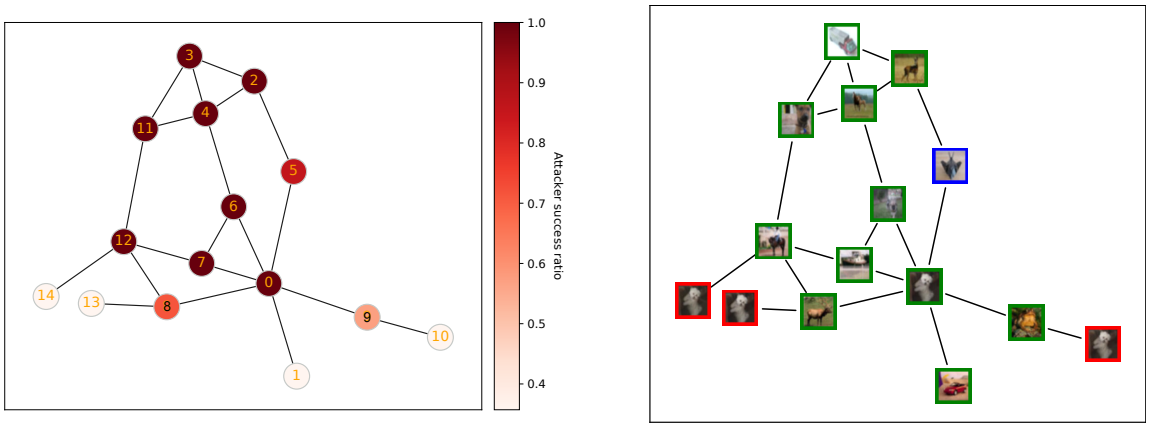



**Figure 4.5** – Reconstruction attacks on D-GD for the Florentine graph (Cifar10, logistic regression model, learning rate $10^{-5}$). Left: the color of each node represents the success rate when that node is the attacker. The success rate is measured as the fraction of nodes for which the reconstructed image achieves a PSNR superior to 10 with respect to the original image (averaged over 10 experiments). Right: detailed view of the case where the attacker is node 5 (highlighted with blue borders). Nodes with green borders are accurately reconstructed, the ones with red borders are not. For completeness, the true input images are shown in Appendix A.7.

expect: even though the gradients of distant nodes are mixed many times before reaching the attacker, our attack allows to disentangle the contributions of the different nodes to enable informative reconstruction up to distance 28. The visual impression is further confirmed by reconstruction metrics over multiple runs (see Figure 4.6).

Finally, we switch to a more complex model for which it is necessary to rely on a gradient inversion attack. We use a small convolutional neural network (see details in Appendix Appendix A.8) on the MNIST dataset and the gradient inversion attack of Geiping et al. [Gei+20a] as a black box. Using a line graph with 31 nodes as in the previous experiment, we see in Figure 4.4(b) that our approach can naturally rely on a black-box gradient inversion attack to reconstruct data from the gradients of more complex models. Here, the reconstructions are accurate up to distance 26. We refer to Appendix Appendix A.7 for results on the Florentine graph.

We note that the performance of our attack on D-GD is sensitive to several parameters. First, having similar local parameters $\theta_v$ across nodes enables better reconstructions as the observed values are primarily influenced by the gradients rather than by variations in the parameters. This condition is easily satisfied either by initializing all the nodes with the same parameters (a standard practice in D-GD) or by waiting until the system approaches convergence. Second, the learning rate plays an important role: it should be small enough to ensure that gradients do not vary wildly across iterations. We illustrate this behavior in Appendix Appendix A.9.

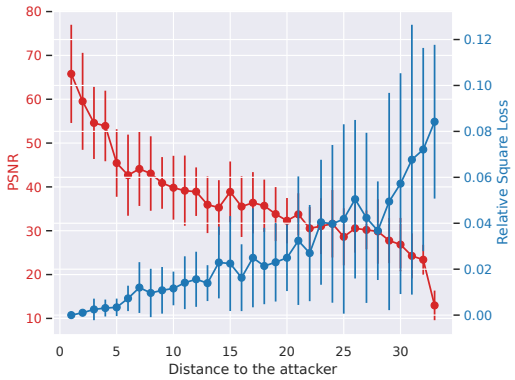



**Figure 4.6** – Reconstruction accuracy (measured by PSNR) and error (measured by the relative square distance) as a function of the distance between the victim and the attacker for D-GD on a line graph (see details in Figure 4.4(a)). The plot shows the mean and standard deviation across 100 experiments.

## 4.6 Conclusion

Our work demonstrates the vulnerability of data when using standard decentralized learning algorithms. More precisely, we show that a node can attack and successfully reconstruct the data of non-neighboring (and sometimes quite distant) nodes by leveraging the communication structure inherent to gossip protocols. We also highlight the impact of the graph topology and the position of the attackers in the success rate of the attack in practice. An interesting open question is a full characterization of reconstructible nodes by structural properties of the graph, which appears to be a challenging problem as we discuss in Appendix A.1. Likewise, optimizing the graph so as to minimize the number of reconstructible nodes is an open question.

Our work shows that one cannot rely on decentralization alone to protect sensitive data. Therefore, to provide robust privacy guarantees, decentralized algorithms must be combined with additional defense mechanisms such as those based on differential privacy. This is the goal of the next chapters. In Chapter 5, we will propose a variant of differential privacy specifically designed for decentralized learning, adopting a trust setting similar to this chapter, where potential attackers are the nodes with their local view on the graph. In Chapter 6, we will use this notion to revisit the privacy analysis of gossip algorithms, but with modifications that enable the derivation of differential privacy guarantees.

# Chapter 5

# Network Differential Privacy

In this chapter, we introduce novel relaxations of local differential privacy (LDP) that naturally arises in fully decentralized algorithms, i.e., when participants exchange information by communicating along the edges of a network graph without central coordinator. These relaxations, that we call network DP and pairwise Network DP, capture the fact that users have only a local view of the system and that the privacy leakage from a user $u$ to a user $v$ may depend on their relative position in the graph for the pairwise version. To show the relevance of network DP, we study a decentralized model of computation where a token performs a walk on a ring and is updated sequentially by the party who receives it. We prove that the privacy-utility trade-offs of our algorithms under network DP significantly improve upon what is achievable under LDP, and often match the utility of the trusted curator model. Our results show for the first time that formal privacy gains can be obtained from full decentralization.

This chapter draws from contributions present in [CB22] and [Cyf+22].

## 5.1 Introduction

In the previous chapter, we illustrated the need to add privacy defenses to decentralized learning to avoid data leakage during the training. To control the privacy leakage, the prominent approach is based on the standard notion of Differential Privacy introduced in detail in Chapter 3. DP guarantees are computed with respect to a trust model.

Several trust models can be considered in in federated and decentralized learning, leading to different privacy-utility trade-offs. The strongest model is local differential privacy (LDP) [Kas+08; DJW13], where each participant (user) does not trust anyone and aims to protect against an adversary that can observe everything that she/he shares. In LDP, random perturbations are performed locally by each user, making it convenient to design private versions of fully decentralized algorithms in this model [see e.g., HMV15; Bel+18; Li+18; Che+19a; XZW20].

Unfortunately, LDP comes at a great cost in utility: for real summation with $n$ users, the best possible error under LDP is a factor $\sqrt{n}$ larger than in the centralized (trusted curator) model of DP [CSS12a], as explained previously in Section 3.2.4. The existing methods to bridge the gap between local and central DP require all users to interact with each other at each step and/or rely on a central coordinator that communicates with all users. Integrating such solutions in fully decentralized algorithms would thus destroy the benefits of full decentralization.

A related line of work has studied mechanisms that "amplify" the DP guarantees of a private algorithm. Beyond privacy amplification by shuffling [Erl+19; Bal+19b; FMT20] (based on the shuffling primitive mentioned above), we can mention amplification by subsampling [BBG18] and amplification by iteration [Fel+18]. These schemes are generally difficult to leverage in a federated/decentralized setting: the former requires that the identity of subsampled participants remain secret, while the latter assumes that only the final result is revealed.

**Our contributions.** In this chapter, we propose a *novel relaxation of LDP where users have only a local view of the decentralized system*, which is a natural assumption in fully decentralized settings. This relaxation, called *Network Differential Privacy*, effectively captures the fact that each user only observes information received from her/his neighbors in the network graph. Network DP can also account for potential collusions between users. We initiate the study of algorithms under network DP in a decentralized model of computation where a token containing the current estimate performs a walk on the network graph and is updated sequentially by the user who receives it. This model has been studied in previous work as a way to perform (non-private) decentralized estimation and optimization with less communication and computation overhead than algorithms that require all users to communicate with their neighbors at each step [RNV09; Mao+20; AR20; JRJ10].

In this rest of the chapter, we illustrate the usefulness of Network Differential Privacy on the simple case of computing real summations and discrete histograms through a deterministic walk over a directed ring. More complex tasks, algorithms and topologies will be studied in the next chapters. For this simple scenario, we propose simple algorithms which achieve a privacy gain of $O(1/\sqrt{n})$ compared to LDP, thereby *matching the privacy-utility trade-off of a trusted aggregator* without relying on any costly secure multi-party computation protocol.

To the best of our knowledge, our work is the first to show that *formal privacy gains can be naturally obtained from full decentralization* (i.e., from having no central coordinator). Our results imply that the true privacy guarantees of some fully decentralized algorithms have been largely underestimated, providing a new incentive for using such approaches beyond the usual motivation of scalability.

The chapter is organized as follows. Section 5.2 introduces our notion of network DP and the decentralized model of computation that we study. Section 5.4 focuses on the case of a

fixed ring topology. Section 5.3 we introduce Pairwise Network Differential Privacy, a variant where privacy budget depends on the pair of nodes.

## 5.2 Network Differential Privacy

Let $V = \{1, \dots, n\}$ be a set of $n$ users (or parties), which are assumed to be honest-but-curious (i.e., they truthfully follow the protocol). Each user $u$ holds a private dataset $D_u$, which we keep abstract at this point. We denote by $D = D_1 \cup \cdots \cup D_n$ the union of all user datasets, and by $D \sim_u D'$ the fact that datasets $D$ and $D'$ of same size differ only on user $u$'s data. This defines a *neighboring relation* over datasets which is sometimes referred to as user-level DP [McM+18]. This relation is weaker than the one used in classic DP and will thus provide stronger privacy guarantees. Indeed, it seeks to hide the influence of a *user's whole dataset* rather than a single of its data points.

We consider a fully decentralized setting, in which users are nodes in a network graph $G = (V, E)$ and an edge $(u, v) \in E$ indicates that user $u$ can send messages to user $v$. The graph may be directed or undirected, and could in principle change over time although we will restrict our attention to fixed topologies. For the purpose of quantifying privacy guarantees, a decentralized algorithm $\mathcal{A}$ will be viewed as a (randomized) mapping which takes as input a dataset $D$ and outputs the transcript of all messages exchanged between users over the network. We denote the (random) output in an abstract manner by $\mathcal{A}(D) = ((u, m, v) :$ user $u$ sent message with content $m$ to user $v$ $)$.

**Network DP.** The key idea of our new relaxation of LDP is to consider that *a given user does not have access to the full transcript $\mathcal{A}(D)$ but only to the messages she/he is involved in* (this can be enforced by the use of secure communication channels). We denote the corresponding view of a user $u$ by

$$\mathcal{O}_u(\mathcal{A}(D)) = ((v, m, v') \in \mathcal{A}(D) : v = u \text{ or } v' = u). \tag{5.1}$$

**Definition 5.1** (Network Differential Privacy)**.** *An algorithm $\mathcal{A}$ satisfies $(\varepsilon, \delta)$-network DP if for all pairs of distinct users $u, v \in V$, all pairs of neighboring datasets $D \sim_u D'$ and all $\mathcal{S} \subset \mathrm{Range}(\mathcal{O}_v(\mathcal{A}))$, we have:*

$$\mathbb{P}(\mathcal{O}_v(\mathcal{A}(D)) \in \mathcal{S}) \leq e^{\varepsilon}\mathbb{P}(\mathcal{O}_v(\mathcal{A}(D')) \in \mathcal{S}) + \delta. \tag{5.2}$$

Network DP essentially requires that for any two users $u$ and $v$, the information gathered by user $v$ during the execution of $\mathcal{A}$ should not depend too much on user $u$'s data. Network DP can be thought of as analyzing the composition of the operator $\mathcal{O}_v$ with the algorithm $\mathcal{A}$. The hope is that in some cases $\mathcal{O}_v \circ \mathcal{A}$ is more private than $\mathcal{A}$: in other words, that applying $\mathcal{O}_v$

*amplifies* the privacy guarantees of $\mathcal{A}$. Note that if $\mathcal{O}_v$ is the identity map (i.e., if each user is able to observe all messages), then Eq. 5.2 boils down to local DP.

We can naturally extend Definition 5.1 to account for potential *collusions* between users. As common in the literature, we assume an upper bound $c$ on the number of users that can possibly collude. The identity of colluders is however unknown to other users. In this setting, we would like to be private with respect to the aggregated information $\mathcal{O}_{V'} = \cup_{v \in V'} \mathcal{O}_v$ acquired by any possible subset $V'$ of $c$ users, as captured by the following generalization of Definition 5.1.

**Definition 5.2** (Network DP with collusions)**.** *An algorithm $\mathcal{A}$ is $(c, \varepsilon, \delta)$-network DP if for each user $u$, all subsets $V' \subset V$ such that $|V'| \leq c$, all pairs of neighboring datasets $D \sim_u D'$, and all $\mathcal{S} \subset \mathrm{Range}(\mathcal{O}_V(\mathcal{A}))$, we have:*

$$\mathbb{P}(\mathcal{O}_{V'}(\mathcal{A}(D)) \in \mathcal{S}) \leq e^{\varepsilon} \mathbb{P}(\mathcal{O}_{V'}(\mathcal{A}(D')) \in \mathcal{S}) + \delta. \tag{5.3}$$

As done previously in Chapter 3, we introduce the variant using Rényi divergence. This definition will be used is most of our results.

**Definition 5.3** (Network Rényi Differential Privacy)**.** *An algorithm $\mathcal{A}$ satisfies $(\alpha, \varepsilon)$-Network Rényi Differential Privacy (NRDP) for $\alpha > 1$ and $\varepsilon > 0$ if for all pairs of neighboring datasets $D \sim D'$:*

$$D_\alpha \left( \mathcal{O}_v(\mathcal{A}(D)) || \mathcal{O}_v(\mathcal{A}(D')) \right) \leq \varepsilon \,. \tag{5.4}$$

## 5.3 Pairwise Network Differential Privacy

In this section, we introduce *pairwise network DP*, a relaxation of LDP which is able to quantify the privacy loss of a decentralized algorithm for each pair of distinct users in a graph. This definition is particularly attractive in situations where nodes want stronger guarantees with respect to some (distant) peers. For instance, in social network graphs, users may have lower privacy expectations with respect to close relatives than regarding strangers. In healthcare, a patient might trust her family doctor more than she trusts other doctors, and in turn more than employees of a regional agency and so on, creating a hierarchical level of trust that our algorithms naturally match.

**Definition 5.4** (Pairwise Network DP)**.** *For $f : \mathcal{V} \times \mathcal{V} \to \mathbb{R}^+$, an algorithm $\mathcal{A}$ satisfies $(\alpha, f)$-Pairwise Network DP (PNDP) if for all pairs of distinct users $u, v \in \mathcal{V}$ and neighboring datasets $D \sim_u D'$:*

$$D_\alpha(\mathcal{O}_v(\mathcal{A}(\mathcal{D})) \,||\, \mathcal{O}_v(\mathcal{A}(\mathcal{D}'))) \leq f(u, v) \,. \tag{5.5}$$

*We note $\varepsilon_{u\to v} = f(u,v)$ the privacy leaked to $v$ from $u$ and say that $u$ is $(\alpha, \varepsilon_{u\to v})$-PNDP with respect to $v$ if only inequality (5.5) holds for $f(u,v) = \varepsilon_{u\to v}$.*

By taking $f$ constant in Definition 5.4, we recover the definition of Network DP (Definition 5.1). Our pairwise variant refines Network DP by allowing the privacy guarantee to depend on $u$ and $v$ (typically, on their relative position in the graph). We refer to Appendix C.7 for a natural adaptation of our definition to the presence of *colluding nodes* and to the protection of *groups* of users in the study of gossip algorithms.

In the rest of this chapter, we will use only Network DP to illustrate the definitions, but Pairwise Network DP will be used in Chapter 6 and 7.

## 5.4 Walk on a Ring

In this section, we recap a very simple random walk algorithm. Then, we analyze a simple special case where the graph is a directed ring, i.e., $E = \{(u, u+1)\}_{u=1}^{n-1} \cup \{(n,1)\}$. The token starts at user $1$ and goes through the ring $K$ times. The ring (i.e., ordering of the nodes) is assumed to be public.

**Decentralized computation model.** We study network DP for decentralized algorithms that perform computations via sequential updates to a *token* $\tau$ walking through the nodes by following the edges of the graph $G$. At each step, the token $\tau$ resides at some node $u$ and is updated by

$$\tau \leftarrow \tau + x_u^k, \quad \text{with } x_u^k = g^k(\tau; D_u), \tag{5.6}$$

where $x_u^k = g^k(\tau; D_u)$ denotes the contribution of user $u$. The notation highlights the fact that this contribution may depend on the current value $\tau$ of the token as well as on the number of times $k$ that the token visited $u$ so far. The token $\tau$ is then sent to another user $v$ for which $(u,v) \in E$.

Provided that the walk follows some properties, this model of computation allows to optimize sums of local cost functions using (stochastic) gradient descent as seen in Section 2.4. In this case, the token $\tau$ holds the model parameters and $x_u^k$ is a (stochastic) gradient of the local loss function of user $u$ evaluated at $\tau$. Such decentralized algorithms can also be used to compute summaries of the users' data, for instance any commutative and associative operation like sums/averages and discrete histograms. In these cases, the contributions of a given user may correspond to different values acquired over time, such as power consumption in smart metering or item ratings in collaborative filtering applications.

**Algorithm 5.1:** Private real summation on the ring

$\tau \leftarrow 0$; $a \leftarrow 0$;
**for** $k = 1$ **to** $K$ **do**
  **for** $u = 1$ **to** $n$ **do**
    **if** $a = 0$ **then**
      $\tau \leftarrow \tau + \text{Perturb}(x_u^k; \sigma_{loc})$;
      $a \leftarrow n - 2$;
    **else**
      $\tau \leftarrow \tau + x_u^k$; $a \leftarrow a - 1$;
**return** $\tau$

### 5.4.1 Real Summation

We first consider the task of estimating the sum $\bar{x} = \sum_{u=1}^{n} \sum_{k=1}^{K} x_u^k$ where the $x$'s are bounded real numbers and $x_u^k$ represents the contribution of user $u$ at round $k$. For this problem, the standard approach in local DP is to add random noise to each single contribution before releasing it. For generality, we consider an abstract mechanism $\text{Perturb}(x; \sigma)$ which adds centered noise with standard deviation $\sigma$ to the contribution $x$ (e.g., the Gaussian or Laplace mechanism). Let $\sigma_{loc}$ be the standard deviation of the noise required so that $\text{Perturb}(\cdot; \sigma_{loc})$ satisfies $(\varepsilon, \delta)$-LDP.

Consider now the simple decentralized protocol in Algorithm 5.1, where noise with the same standard deviation $\sigma_{loc}$ is added *only once every* $n - 1$ *hops of the token*. By leveraging the fact that the view of each user $u$ is restricted to the values taken by the token at each of its $K$ visits to $u$, combined with advanced composition [DRV10], we have the following result (see Appendix B.1 for the proof).

**Theorem 5.5.** *Let $\varepsilon, \delta > 0$. Algorithm 5.1 outputs an unbiased estimate of $\bar{x}$ with standard deviation $\sqrt{\lfloor Kn/(n-1) \rfloor}\sigma_{loc}$, and is $(\sqrt{2K \log(1/\delta')}\varepsilon + K\varepsilon(e^{\varepsilon} - 1), K\delta + \delta')$-network DP for any $\delta' > 0$.*

To match the same privacy guarantees, LDP incurs a standard deviation of $\sqrt{Kn}\sigma_{loc}$. Therefore, Algorithm 5.1 provides an $O(1/\sqrt{n})$ reduction in error or, equivalently, an $O(1/\sqrt{n})$ gain in $\varepsilon$. In fact, Algorithm 5.1 achieves the same privacy-utility trade-off as a *trusted central aggregator* that would iteratively aggregate the raw contributions of all users at each round $k$ and perturb the result before sending it to the users, as done in federated learning algorithms with a trusted server [Kai+21].

**Remark 5.6.** *We can design variants of Algorithm 5.1 in which noise addition is distributed across users. Using the Gaussian mechanism, each user can add noise with std. dev. $\sigma'_{loc} = \sigma_{loc}/\sqrt{n}$,*

**Algorithm 5.2:** Private histogram on the ring

Init. $\tau \in \mathbb{N}^L$ with $\gamma n$ random elements
**for** $k = 1$ **to** $K$ **do**
  **for** $u = 1$ **to** $n$ **do**
    $y_u^k \leftarrow RR_\gamma(x_u^k)$
    $\tau[y_u^k] \leftarrow \tau[y_u^k] + 1$
**for** $i = 0$ **to** $L - 1$ **do**
  $\tau[i] \leftarrow \frac{\tau[i] - \gamma/L}{1-\gamma}$
**return** $\tau$

*except for the very first contribution which requires std. dev. $\sigma_{loc}$ to properly hide the contributions of users in the first cycle. The total added noise has std. dev. $\sqrt{\lfloor Kn/(n-1) \rfloor + 1}\sigma_{loc}$, leading to same utility as Algorithm 5.1 (up to a constant factor that is negligible when $K$ is large).*

### 5.4.2 Discrete Histogram Computation

We now turn to the computation of histograms over a discrete domain $[L] = \{1, \dots, L\}$. The goal is to compute $h \in \mathbb{N}^L$ s.t. $h_l = \sum_{u=1}^{n} \sum_{k=1}^{K} \mathbb{I}[x_u^k = l]$, where $x_u^k \in [L]$. A classic approach in LDP is based on $L$-ary randomized response [KOV14], where each user submits its true value with probability $1 - \gamma$ and a uniformly random value with probability $\gamma$. We denote this primitive by $RR_\gamma : [L] \to [L]$.

In our setting with a ring network, we propose Algorithm 5.2, where each contribution of a user is randomized using $RR_\gamma$ before being added to the token $\tau \in \mathbb{N}^L$. Additionally, $\tau$ is initialized with enough random elements to hide the first contributions. Note that at each step, the token contains a partial histogram equivalent to a shuffling of the contributions added so far, allowing us to leverage results on *privacy amplification by shuffling* [Erl+19; Bal+19b; FMT20]. In particular, we can prove the following utility and privacy guarantees for Algorithm 5.2 (see Appendix B.2 for the proof).

**Theorem 5.7.** *Let $\varepsilon < \frac{1}{2}$, $\delta \in (0, \frac{1}{100})$, and $n > 1000$. Let $\gamma = L/(\exp(12\varepsilon\sqrt{\log(1/\delta)/n}) + L - 1)$. Algorithm 5.2 outputs an unbiased estimate of the histogram with $\gamma n(K+1)$ expected random responses. Furthermore, it satisfies $(\sqrt{2K\log(1/\delta')}\varepsilon + K\varepsilon(e^\varepsilon - 1), K\delta + \delta')$-network DP for any $\delta' > 0$.*

Achieving the same privacy in LDP would require $\gamma$ to be constant in $n$, hence $\sqrt{n}$ times more random responses. Equivalently, if we fix utility (i.e., $\gamma$), Theorem 5.7 shows that Algorithm 5.2 again provides a privacy gain of $\frac{1}{n}\sqrt{n/\log(1/\delta)} = O(1/\sqrt{n})$ compared to LDP.

**Remark 5.8.** *For clarity, Theorem 5.7 relies on the amplification by shuffling result of [Erl+19] which has a simple closed form. A tighter and more general result (with milder restrictions on the values of $n$, $\varepsilon$ and $\delta$) can be readily obtained by using the results of [Bal+19b] and [FMT20].*

**Remark 5.9.** *Algorithm 5.1 (real summation) can also be used to perform histogram computation. However, for domains of large cardinality $L$ (e.g., $L \gg n$), Algorithm 5.2 requires fewer random numbers and maintains a sparse (more compact) representation of the histogram.*

### 5.4.3 Discussion

We have seen that decentralized computation over a ring provides a simple way to achieve utility similar to a trusted aggregator thanks to the sequential communication that hides the contribution of previous users in a summary. We emphasize that this is achieved without relying on a central server (only local communications) or resorting to costly multi-party computation protocols (only two secure communication channels per user are needed). Interestingly, the ring topology is often used in practical deployments and theoretical analysis of (non-private) decentralized algorithms [Lia+17; Tan+18; Kol+20; Neg+20; Mar+20], owing to its simplicity and good empirical performance. Finally, we note an interesting connection between the case of network DP over a ring topology and the pan-privacy model for streaming algorithms [Dwo+10] (see Appendix B.3 for details).

Despite the above advantages, the use of a fixed ring topology has some limitations. First, the above algorithms are not robust to collusions: in particular, if two users collude and share their view, Algorithm 5.1 does not satisfy DP. While this can be mitigated by distributing the noise addition across users (Remark 5.6), a node placed between two colluding nodes (or with few honest users in-between) would suffer largely degraded privacy guarantees. A similar reasoning holds for Algorithm 5.2. Second, a fixed ring topology is not well suited to extensions to gradient descent, where we would like to leverage privacy amplification by iteration [Fel+18]. In the latter, the privacy guarantee for a given user (data point) grows with the number of gradient steps that come after it. In a fixed ring, the privacy of a user $u$ with respect to another user $v$ would thus depend on their relative positions in the ring (e.g., there would be no privacy amplification when $v$ is the user who comes immediately after $u$). These limitations motivate us to consider random walks on a complete graph and on arbitrary topologies that are studied in Chapter 7.

## 5.5 Conclusion

In this chapter, we focus on the definition of two relaxations of differential privacy, which impose privacy constraints based on a node's view. This view corresponds to either a subset or a projection of the entire set of outputs from the algorithm. These relaxations introduce interesting trust models that notably emphasize the dependence of privacy guarantees on the graph structure. In the next chapter, we will explore these models in more detail, first in the context of gossip algorithms (Chapter 6), then random walks (Chapter 7), and Alternating Direction Method of Multipliers (ADMM) algorithms in Chapter 8.

# Chapter 6

# Muffliato: Peer-to-Peer Privacy Amplification in Gossip

In this chapter, we analyze, under the Pairwise Network DP setting introduced in Chapter 5, the combination of local noise injection with (simple or randomized) gossip averaging protocols on fixed and random communication graphs. We also derive a differentially private decentralized optimization algorithm that alternates between local gradient descent steps and gossip averaging. Our results show that our algorithms amplify privacy guarantees as a function of the distance between nodes in the graph, matching the privacy-utility trade-off of the trusted curator, up to factors that explicitly depend on the graph topology. Remarkably, these factors become constant for expander graphs. Finally, we illustrate our privacy gains with experiments on synthetic and real-world datasets.

This chapter is drawn from [Cyf+22].

## 6.1 Introduction

In this chapter, we use the privacy model defined in Chapter 5 to formally quantify the privacy amplification for the fundamental brick of communication at the core of decentralized optimization: gossip algorithms. Calling *Muffliato* the combination of local noise injection with a gossip averaging protocol, we precisely track the resulting privacy leakage between each pair of nodes. Through gossiping, the private values and noise terms of various users add up, obfuscating their contribution well beyond baseline LDP guarantees: as their distance in the graph increases, the privacy loss decreases. We then show that the choice of graph is crucial to enforce a good privacy-utility trade-off while preserving the scalability of gossip algorithms.

(***i***) We propose *Muffliato*[1], a privacy amplification mechanism composed of local Gaussian noise injection at the node level followed by gossiping for averaging the private values. It offers privacy amplification that increases as the distance between two nodes increases. Informally, the locally differentially private value shared by a node $u$ is mixed with other contributions, to the point that the information that leaks to another node $v$ can have a very small sensitivity to the initial value in comparison to the accumulated noise. Our contributions can be described as following.

(***ii***) We analyze both synchronous gossip [Dim+10] and randomized gossip [Boy+06] under a unified privacy analysis with arbitrary time-varying gossip matrices. We show that the magnitude of the privacy amplification is significant: the average privacy loss over all the pairs in this setting reaches the optimal privacy-utility trade-off of a trusted aggregator, up to a factor $\frac{d}{\sqrt{\lambda_W}}$, where $\lambda_W$ is the weighted graph eigengap and $d$ the maximum degree of the graph. Remarkably, this factor can be of order 1 for expanders, yielding a sweet spot in the privacy-utility-scalability trade-off of gossip algorithms. Then, we study the case where the graph is itself random and private, and derive stronger privacy guarantees.

(***iii***) Finally, we develop and analyze differentially private decentralized Gradient Descent (GD) and Stochastic Gradient Descent (SGD) algorithms to minimize a sum of local objective functions. Building on *Muffliato*, our algorithms alternate between rounds of differentially private gossip communications and local gradient steps. We prove that they enjoy the same privacy amplification described above for averaging, up to factors that depend on the regularity of the global objective.

(***iv***) We demonstrate the usefulness of our approach and analysis through experiments on synthetic and real-world datasets and network graphs. We illustrate how privacy is amplified between nodes in the graph as a function of their distance, and show how time-varying random graphs can be used to split the privacy loss more uniformly across nodes in decentralized optimization.

## 6.2 Private Gossip Averaging

In this section, we analyze a generic algorithm with arbitrary time-varying communication matrices for averaging. Then, we instantiate and discuss these results for synchronous communications with a fixed gossip matrix, communications using randomized gossip [Boy+06], and with Erdös-Rényi graphs, whose non-private versions were introduced in Chapter 2.

[1]The name is borrowed from the Harry Potter series: it designates a "spell that filled the ears of anyone nearby with an unidentifiable buzzing", thereby concealing messages from unintended listeners through noise injection.

### 6.2.1 General Privacy Analysis of Gossip Averaging

We consider gossip over time-varying graphs $(G_t)_{0\leq t\leq T}$, defined as $G_t = (\mathcal{V}, \mathcal{E}_t)$, with corresponding gossip matrices $(W_t)_{0\leq t\leq T}$. The *generic Muffliato* algorithm $\mathcal{A}^T$ for averaging $x = (x_v)_{v\in\mathcal{V}}$ corresponds to an initial noise addition followed by $T$ gossip steps, similarly to non-private version defined in Section 2.4.1. Writing $W_{0:t} = W_{t-1}\dots W_0$, the iterates of $\mathcal{A}^T$ are thus defined by:

$$\forall v \in \mathcal{V}, x_v^0 = x_v + \eta_v \text{ with } \eta_v \sim \mathcal{N}\left(0, \sigma^2\right), \quad \text{and } x^{t+1} = W_t x^t = W_{0:t+1}(x+\eta)\,. \tag{6.1}$$

Note that the update rule at node $v \in \mathcal{V}$ writes as $x_v^{t+1} = \sum_{w\in\mathcal{N}_t(v)}(W_t)_{v,w}x_w^t$ where $\mathcal{N}_t(v)$ are the neighbors of $v$ in $G_t$, so for the privacy analysis, the view of a node is:

$$\mathcal{O}_v\left(\mathcal{A}^T(\mathcal{D})\right) = \{(W_{0:t}(x+\eta))_w \mid \quad \{v,w\}\in\mathcal{E}_t\,, \quad 0\leq t\leq T-1\}\cup\{x_v\}\,. \tag{6.2}$$

**Theorem 6.1.** *Let $T \geq 1$ and denote by $\mathcal{P}^T_{\{v,w\}} = \{s < T : \{v,w\}\in\mathcal{E}_s\}$ the set of time-steps with communication along edge $\{v,w\}$. Then, $\mathcal{A}^T$ is $(\alpha, f)$-PNDP with:*

$$f(u,v) = \frac{\alpha\Delta^2}{2\sigma^2}\sum_{w\in\mathcal{V}}\sum_{t\in\mathcal{P}^T_{\{v,w\}}}\frac{(W_{0:t})^2_{u,w}}{\|(W_{0:t})_w\|^2}\,. \tag{6.3}$$

This theorem, proved in Appendix C.2, gives a tight computation of the privacy loss between every pair of nodes and can easily be computed numerically (see Section 6.4). Since distant nodes correspond to small entries in $W_{0:t}$, Equation 6.3 suggests that they reveal less to each other. We will characterize this precisely for the case of fixed communication graph in the next subsection.

In addition to pairwise guarantees, we can interpret the result of Theorem 6.1 by introducing the *mean privacy loss* $\bar{\varepsilon}_v = \frac{1}{n}\sum_{u\in\mathcal{V}\setminus\{v\}} f(u,v)$. It allows us to compare NDP guarantees with baselines LDP and trusted aggregator by enforcing $\bar{\varepsilon} = \max_{v\in\mathcal{V}}\bar{\varepsilon}_v \leq \varepsilon$. The value $\bar{\varepsilon}_v$ is the average of the privacy loss from all the nodes to $v$ and thus does not correspond to a proper privacy guarantee, but it provides a convenient way to summarize our gains, noting that distant nodes — in ways that will be specified — will have better privacy guarantee than this average, while worst cases will remain bounded by the baseline LDP guarantee provided by local noise injection. For Theorem 6.1, the mean privacy satisfies:

$$\bar{\varepsilon}_v = \frac{\alpha\Delta^2 T_v}{2n\sigma^2}\,, \tag{6.4}$$

where $T_v$ is the total number of communications node $v$ was involved with up to time $T$. Thus, in comparison with LDP, the mean privacy towards $v$ is $n/T_v$ times smaller. In other words, a node learns much less than in LDP as long as it communicates $o(n)$ times.

**Algorithm 6.1:** MUFFLIATO

**Input:** local values $(x_v)_{v\in\mathcal{V}}$ to average, gossip matrix $W$ on a graph $G$, in $T$ iterations, noise variance $\sigma^2$
$\gamma \leftarrow 2\frac{1-\sqrt{\lambda_W(1-\frac{\lambda_W}{4})}}{(1-\lambda_W/2)^2}$
**for** *all nodes* $v$ *in parallel* **do**
  $x_v^0 \leftarrow x_v + \eta_v$ where $\eta_v \sim \mathcal{N}(0,\sigma^2)$
**for** $t = 0$ *to* $T-1$ **do**
  **for** *all nodes* $v$ *in parallel* **do**
    **for** *all neighbors* $w$ *defined by* $W$ **do**
      Send $x_v^t$, receive $x_w^t$
    $x_v^{t+1} \leftarrow (1-\gamma)x_v^{t-1} + \gamma\sum_{w\in\mathcal{N}_v} W_{v,w}x_w^t$

### 6.2.2 Private Synchronous *Muffliato*

We now consider *Muffliato* over a fixed graph (Algorithm 6.1). Note that we use gossip acceleration (see Definition 2.4). We start by analyzing the utility of *Muffliato*, which decomposes as an averaging error term vanishing exponentially fast, and a *bias* term due to the noise. General convergence rates are given in Appendix C.3, from which we extract the following result.

**Theorem 6.2** (Utility analysis)**.** *Let $\lambda_W$ be the spectral gap of $W$. Muffliato (Algorithm 6.1) verifies, for any $t \geq T^{\mathrm{stop}}$:*

$$\frac{1}{2n}\sum_{v\in\mathcal{V}}\mathbb{E}\left(\left\|x_v^t-\overline{x}\right\|^2\right) \leq \frac{3\sigma^2}{n}, \quad \textit{where} \quad T^{\mathrm{stop}} = \frac{1}{\sqrt{\lambda_W}}\log\left(\frac{n}{\sigma^2}\max\left(\sigma^2, \frac{1}{n}\sum_{v\in\mathcal{V}}\|x_v-\overline{x}\|^2\right)\right).$$

For the privacy guarantees, Theorem 6.1 still holds as accelerated gossip can be seen as a post-processing of the non-accelerated version. Thanks to the fixed graph, we can derive a more explicit formula.

**Corollary 6.3.** *Algorithm 6.1 satisfies $(\alpha, \varepsilon_{u\to v}^T(\alpha))$-PNDP for node $u$ with respect to $v$, with:*

$$\varepsilon_{u\to v}^T(\alpha) \leq \frac{\alpha\Delta^2 n}{2\sigma^2}\max_{\{v,w\}\in\mathcal{E}} W_{v,w}^{-2}\sum_{t=1}^{T}\mathbb{P}(X^t=v|X^0=u)^2,$$

**Algorithm 6.2:** RANDOMIZED MUFFLIATO

**Input:** local values $(x_v)_{v\in\mathcal{V}}$ to average, activation intensities $(p_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$, in $T$ iterations, noise variance $\sigma^2$
**for** *all nodes* $v$ *in parallel* **do**
  $x_v^0 \leftarrow x_v + \eta_v$ where $\eta_v \sim \mathcal{N}(0, \sigma^2)$
**for** $t = 0$ *to* $T-1$ **do**
  Sample $\{v_t, w_t\} \in \mathcal{E}$ with probability $p_{\{v_t,w_t\}}$
  $v_t$ and $w_t$ exchange $x_{v_t}^t$ and $x_{w_t}^t$
  Local averaging: $x_{v_t}^{t+1} = x_{w_t}^{t+1} = \frac{x_{v_t}^{t+1} + x_{w_t}^{t+1}}{2}$
  For $v \in \mathcal{V} \setminus \{v_t, w_t\}$, $x_v^{t+1} = x_v^t$

**Table 6.1** – Utility of *Muffliato* for several topologies under the constraint $\overline{\varepsilon} \leq \varepsilon$ for the classic gossip matrix where $W_{v,w} = \min(1/d_v, 1/d_w)$ and $d_v$ is the degree of node $v$. $\tilde{\mathcal{O}}(\cdot)$ hides constant and logarithmic factors. Recall that utility is $\tilde{\mathcal{O}}(\alpha\Delta^2/n\varepsilon)$ for LDP and $\tilde{\mathcal{O}}(\alpha\Delta^2/n^2\varepsilon)$ for central DP.

| **Graph** | Arbitrary | Expander | $C$-Torus | Complete | Ring |
|---|---|---|---|---|---|
| Algorithm 6.1 | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2 d}{n^2\varepsilon\sqrt{\lambda_W}}\right)$ | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n^2\varepsilon}\right)$ | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2 C}{n^{2-1/C}\varepsilon}\right)$ | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n\varepsilon}\right)$ | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n\varepsilon}\right)$ |
| Algorithm 6.2 | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n^2\varepsilon\lambda_W}\right)$ | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n^2\varepsilon}\right)$ | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n^{2-2/C}\varepsilon}\right)$ | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n^2\varepsilon}\right)$ | $\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n\varepsilon}\right)$ |

> *where $(X^t)_t$ is the random walk on graph $G$, with transitions $W$.*

This result allows us to directly relate the privacy loss from $u$ to $v$ to the probability that the random walk on $G$ with transition probabilities given by the gossip matrix $W$ goes from $u$ to $v$ in a certain number of steps. It thus captures a notion of distance between nodes in the graph. We also report the utility under fixed mean privacy loss $\overline{\varepsilon} = \max_{v\in\mathcal{V}} \overline{\varepsilon}_v \leq \varepsilon$ in Table 6.1 for various graphs introduced in Section 2.4, where one can see a utility-privacy trade-off improvement of $n\sqrt{\lambda_W}/d$, where $d$ is the maximum degree, compared to LDP. Using expanders closes the gap with a trusted aggregator (i.e., central DP) up to constant and logarithmic terms. Remarkably, graph topologies that make gossip averaging efficient (i.e. with big $\sqrt{\lambda_W}/d$), such as exponential graphs or hypercubes [Yin+21], are also the ones that achieve optimal privacy amplification (up to logarithmic factors). In other words, *privacy, utility and scalability are compatible*.

### 6.2.3 Private Randomized *Muffliato*

Synchronous protocols require global coordination between nodes, which can be costly or even impossible in some settings. On the contrary, asynchronous protocols only require separated activation of edges: they are thus are more resilient to stragglers nodes and faster in practice. In asynchronous gossip, at a given time-step a single edge $\{u, v\}$ is activated independently from the past with probability $p_{\{u,v\}}$, as described in Section 2.4 and in particular Section 2.4.1.

In our setting, randomized *Muffliato* (Algorithm 6.2) corresponds to instantiating our general analysis with $W^t = W_{\{v_t,w_t\}} = I_n - (e_{v_t} - e_{w_t})(e_{v_t} - e_{w_t})^\top/2$ if $\{v_t, w_t\}$ is sampled at time $t$. The utility analysis is similar to the synchronous case.

**Theorem 6.4** (Utility analysis). *Let $\lambda(p)$ be the spectral gap of graph $G$ with weights $(p_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$. Randomized* Muffliato (*Algorithm 6.2*) *verifies, for all $t \geq T^{\mathrm{stop}}$:*

$$\frac{1}{2n}\sum_{v\in\mathcal{V}} \mathbb{E}\left\|x_v^t - \overline{x}\right\|^2 \leq \frac{2\sigma^2}{n}, \quad \textit{where } T^{\mathrm{stop}} = \frac{1}{\lambda(p)}\log\left(\frac{n}{\sigma^2}\max\left(\sigma^2, \frac{1}{n}\sum_{v\in\mathcal{V}}\left\|x_v^0 - \overline{x}\right\|^2\right)\right).$$

To compare with synchronous gossip (Algorithm 6.1), we note that activation probabilities can be derived from a gossip matrix $W$ by taking $p_{\{u,v\}} = 2W_{\{u,v\}}/n$ implying that $\lambda(p) = 2\lambda_W/n$, thus requiring $n$ times more iterations to reach the same utility as the synchronous applications of matrix $W$. However, for a given time-horizon $T$ and node $v$, the number of communications $v$ can be bounded with high probability by a $T/n$ multiplied by a constant whereas Algorithm 6.1 requires $d_v T$ communications. Consequently, as reported in Table 6.1, for a fixed privacy mean $\overline{\varepsilon}_v$, Algorithm 6.2 has the same utility as Algorithm 6.1, up to two differences: the degree factor $d_v$ is removed, while $\sqrt{\lambda_W}$ degrades to $\lambda_W$ as we do not accelerate randomized gossip (see Remark 6.5 below). Randomized gossip can thus achieve an optimal privacy-utility trade-off with large-degree graphs, as long as the spectral gap is small enough.

**Remark 6.5** (Accelerating Randomized *Muffliato*). *For simplicity, Randomized* Muffliato (*Algorithm 6.2*) *is not accelerated, while* Muffliato (*Algorithm 6.1*) *uses Chebychev acceleration to obtain a dependency on $\sqrt{\lambda_W}$ rather than $\lambda_W$ (see Chapter 2 for definition). Thus, and as illustrated by Table 6.1, Algorithm 6.2 does not improve over Algorithm 6.1 for all values of $d$ (maximum degree), $n$ and $\lambda_W$. However, Algorithm 6.2 can be accelerated using a continuized version of Nesterov acceleration [EHM20; Eve+21], thus replacing $\lambda(p)$ in the expression of $T^{\mathrm{stop}}$ in Theorem 6.4, by $\sqrt{\lambda(p)/(dn)}$. Doing so, using randomized communications improve privacy guarantees over Algorithm 6.1 for all graphs considered in Table 6.1.*

### 6.2.4 Erdös-Rényi Graphs

So far the graph was considered to be public and the amplification only relied on the secrecy of the messages. In practice, the graph may be sampled randomly and the nodes need only to know their direct neighbors. We show that we can leverage this through the weak convexity of Rényi DP to amplify privacy between non-neighboring nodes. We focus on Erdös-Rényi graphs, which can be built without central coordination by picking each edge independently with the same probability $q$. For $q = c\log(n)/n$ where $c > 1$, Erdös-Rényi graphs are good expanders with node degrees $d_v = \mathcal{O}(\log n)$ and $\lambda_W$ concentrating around 1 [HKP19]. We obtain the following privacy guarantees.

**Theorem 6.6** (*Muffliato* on a random Erdös-Rényi graph)**.** *Let us fix $u, v \in \mathcal{V}$ distinct nodes. Let $\alpha > 1$, $T \geq 0$, $\sigma^2 \geq \frac{\Delta^2 \alpha(\alpha-1)}{2}$ and $q = c\frac{\log(n)}{n}$ for $c > 1$. After running Algorithm 6.1 with these parameters, node $u$ is $(\alpha, \varepsilon^T_{u\to v}(\alpha))$-PNDP with respect to $v$, with:*

$$\varepsilon^T_{u\to v}(\alpha) \leq \begin{cases} \dfrac{\alpha\Delta^2}{2\sigma^2} & \text{with probability } q\,, \\ \dfrac{\alpha\Delta^2}{\sigma^2}\dfrac{Td_v}{n-d_v} & \text{with probability } 1-q\,. \end{cases}$$

This results shows that with probability $q$, $u$ and $v$ are neighbors and there is no amplification compared to LDP. The rest of the time, with probability $1-q$, the privacy matches that of a trusted aggregator up to a degree factor $d_v = \mathcal{O}(\log n)$ and $T = \tilde{\mathcal{O}}(1/\sqrt{\lambda_W}) = \tilde{\mathcal{O}}(1)$ [HKP19]. In particular, if several rounds of gossip averaging are needed, as in the next section for SGD, changing the graph mitigates the privacy loss of the rounds where two nodes are neighbors thanks to the rounds where they are not.

## 6.3 Private Decentralized Optimization

We now build upon *Muffliato* to design decentralized optimization algorithms. Each node $v \in \mathcal{V}$ possesses a data-dependent function $\phi_v : \mathbb{R}^D \to \mathbb{R}$ and we wish to *privately* minimize the function

$$\phi(\theta) = \frac{1}{n}\sum_{v\in\mathcal{V}} \phi_v(\theta)\,, \quad \text{with } \phi_v(\theta) = \frac{1}{|\mathcal{D}_v|}\sum_{x_v\in\mathcal{D}_v} \ell_v(\theta, x_v)\,, \quad \theta \in \mathbb{R}^D\,, \tag{6.5}$$

where $\mathcal{D}_v$ is the (finite) dataset corresponding to user $v$ for data lying in a space $\mathcal{X}_v$, and $\ell_v : \mathbb{R}^D \times \mathcal{X}_v \to \mathbb{R}$ a loss function. We assume that $\phi$ is $\mu$-strongly convex, and each $\phi_v$ is $L$-smooth, and denote $\kappa = L/\mu$. We note that our results can be extended to the general convex and smooth setting. Denoting by $\theta^\star$ the minimizer of $\phi$, for some non-negative $(\zeta_v^2)_{v\in\mathcal{V}}$, $(\rho_v^2)_{v\in\mathcal{V}}$ and all $v \in \mathcal{V}$, we assume:

$$\|\nabla\phi_v(\theta^\star) - \nabla\phi(\theta^\star)\|^2 \leq \zeta_v^2 \quad , \quad \mathbb{E}(\|\nabla\ell_v(\theta^\star, x_v) - \nabla\phi(\theta^\star)\|^2) \leq \rho_v^2\,, \quad x_v \sim \mathcal{L}_v\,,$$

where $\mathcal{L}_v$ is the uniform distribution over $\mathcal{D}_v$. We write $\bar{\rho}^2 = \frac{1}{n}\sum_{v\in\mathcal{V}}\rho_v^2$ and $\bar{\zeta}^2 = \frac{1}{n}\sum_{v\in\mathcal{V}}\zeta_v^2$.

We introduce Algorithm 6.3, a private version of the classical decentralized SGD algorithm studied in [Kol+20]. Inspired by the optimal algorithm MSDA of Scaman et al. [Sca+17] that alternates between $K$ Chebychev-accelerated gossip communications and expensive dual gradient computations, our Algorithm 6.3 alternates between $K$ Chebychev-accelerated gossip communications and cheap local stochastic gradient steps. This alternation reduces the total number of gradients leaked, a crucial point for achieving good privacy. Note that in Algorithm 6.3, each communication round uses a potentially different gossip matrix $W_t$. In the

**Algorithm 6.3:** Muffliato-SGD and Muffliato-GD

**Input:** initial points $\theta_v^0 \in \mathbb{R}^D$, number of iterations $T$, step sizes $\nu > 0$, noise variance $\sigma^2$, gossip matrices $(W_t)_{t\geq 0}$, local functions $\phi_v$, number of communication rounds $K$
**for** $t = 0$ *to* $T-1$ **do**
  **for** *all nodes* $v$ *in parallel* **do**
    Compute $\hat{\theta}_v^t = \theta_v^t - \nu \nabla_\theta \ell_v(\theta_v^t, x_v^t)$ where $x_v^t \sim \mathcal{L}_v$
  $\theta_v^{t+1} = \textsc{Muffliato}\left((\hat{\theta}_v^t)_{v\in\mathcal{V}}, W_t, K, \nu^2\sigma^2\right)$

results stated below, we fix $W_t = W$ for all $t$ and defer the more general case to Appendix C.6, where different independent Erdös-Rényi graphs with same parameters are used at each communication round.

**Remark 6.7.** *Our setting encompasses both GD and SGD.* Muffliato-*GD is obtained by removing the stochasticity,* i.e., *setting* $\ell_v(\cdot) = \phi_v(\cdot)$. *In that case,* $\bar{\rho}^2 = 0$.

**Theorem 6.8** (Utility analysis of Algorithm 6.3)**.** *For suitable step-size parameters, for a total number of* $T^{\mathrm{stop}}$ *computations and* $T^{\mathrm{stop}}K$ *communications, with:*

$$T^{\mathrm{stop}} = \tilde{\mathcal{O}}\left(\kappa\right), \quad \text{and} \quad K = \left\lceil \sqrt{\lambda_W}^{-1} \log\left(\max\left(n, \frac{\bar{\zeta}^2}{D\sigma^2 + \bar{\rho}^2}\right)\right)\right\rceil,$$

*the iterates* $(\theta^t)_{t\geq 0}$ *generated by Algorithm 6.3 verify* $\mathbb{E}(\phi(\tilde{\theta}^{\mathrm{out}}) - \phi(\theta^\star)) = \tilde{\mathcal{O}}(\frac{D\sigma^2+\bar{\rho}^2}{\mu n T^{\mathrm{stop}}})$ *where* $\tilde{\theta}^{\mathrm{out}} \in \mathbb{R}^D$ *is a weighted average of the* $\bar{\theta}^t = \frac{1}{n}\sum_{v\in\mathcal{V}} \theta_v^t$ *until* $T^{\mathrm{stop}}$.

For the following privacy analysis, we need a bound on the sensitivity of gradients with respect to the data. To this end, we assume that for all $v$ and $x_v$, $\ell_v(\cdot, x_v)$ is $\Delta_\phi/2$ Lipschitz[2].

**Theorem 6.9** (Privacy analysis of Algorithm 6.3)**.** *Let* $u$ *and* $v$ *be two distinct nodes in* $\mathcal{V}$. *After* $T$ *iterations of Algorithm 6.3 with* $K \geq 1$, *node* $u$ *is* $(\alpha, \varepsilon_{u\to v}^T(\alpha))$*-PNDP with respect to* $v$, *with:*

$$\varepsilon_{u\to v}^T(\alpha) \leq \frac{T^2\Delta_\phi^2\alpha}{2\sigma^2} \sum_{k=0}^{K-1} \sum_{w:\{v,w\}\in\mathcal{E}} \frac{(W^k)_{u,w}^2}{\|(W^k)_w\|^2}. \tag{6.6}$$

*Thus, for any* $\varepsilon > 0$, *Algorithm 6.3 with* $T^{\mathrm{stop}}(\kappa, \sigma^2, n)$ *steps and for* $K$ *as in Theorem 6.8, there exists* $f$ *such that the algorithm is* $(\alpha, f)$*-pairwise network DP, with:*

$$\forall v \in \mathcal{V}, \quad \bar{\varepsilon}_v \leq \varepsilon \quad \text{and} \quad \mathbb{E}\phi(\tilde{\theta}^{\mathrm{out}}) - \phi(\theta^\star)) \leq \tilde{\mathcal{O}}\left(\frac{\alpha D\Delta_\phi^2\kappa d}{\mu n^2\varepsilon\sqrt{\lambda_W}} + \frac{\bar{\rho}^2}{nL}\right),$$

[2]This assumption can be replaced by the more general Assumption C.8 given in Appendix C.6.

*where* $d = \max_{v \in \mathcal{V}} d_v$.

The term $\frac{\bar{\rho}^2}{nL}$ above (which is equal to zero for *Muffliato*-GD, see Remark 6.7) is privacy independent. It is typically dominated by the first term, which corresponds to the utility loss due to privacy. Comparing Theorem 6.9 with the results for *Muffliato* (Table 6.1 in Section 6.2.2), the only difference lies in the factor $D\Delta_\phi^2\kappa/\mu$. Note that $\Delta_\phi^2$ plays the role of the sensitivity $\Delta^2$, and $D$ appears naturally due to considering $D$-dimensional parameters. On the other hand, $\kappa/\mu$ is directly related to the complexity of the optimization problem: the easier the problem, the better the privacy-utility trade-off of our algorithm. Regarding the influence of the graph, the same discussion as after Corollary 6.3 applies here. In particular, for expander graphs like the exponential graph of [Yin+21], the factor $d/\sqrt{\lambda_W}$ is constant. In this case, converting to standard $(\varepsilon, \delta)$-DP gives $\tilde{\mathcal{O}}\left(\frac{D\Delta_\phi^2\kappa}{\mu n^2 \varepsilon^2}\right)$, recovering the optimal privacy-utility trade-off of central DP [BST14; WYX17] up to a factor $\kappa$. Note that we achieve this nearly optimal rate under a near-linear gradient complexity of $T^{\text{stop}}(\kappa, \sigma^2, n)n = \tilde{\mathcal{O}}(\kappa n)$ and near-linear total number of messages $T^{\text{stop}}(\kappa, \sigma^2, n)Kn = \tilde{\mathcal{O}}(\kappa n)$.

**Remark 6.10** (Time-varying graphs)**.** *The analysis of* Muffliato*-GD/SGD presented in this section (Theorems 6.8 and 6.9) assumes constant gossip matrices* $W_t = W$*. A more general version of these results is presented in Appendix C.6 to handle any fixed sequence of matrices* $(W_t)_t$ *and graphs. This can be used to model randomized communications (as previously described for gossip averaging in Section 6.2.3) as well as user dropout (see experiments in Appendix C.8). Time-varying graphs can also be used to split the privacy loss more uniformly across the different nodes by avoiding that nodes have the same neighbors across multiple gossip computations. We illustrate this experimentally for decentralized optimization in Section 6.4, where we change the graph after each gradient step of* Muffliato*-GD. Note that our analysis does not allow to select the next graph adaptively to the current privacy losses, as this would require other analysis techniques, such as privacy odometers [Rog+16; FZ20; KTH22].*

## 6.4 Experiments

In this section, we show that pairwise network DP provides significant privacy gains in practice even for moderate size graphs. We use synthetic graphs and real-world graphs for gossip averaging. For decentralized optimization, we solve a logistic regression problem on real-world data with time-varying Erdos-Rényi graphs, showing in each case clear gains in privacy compared to LDP. The code used to obtain these results is available at https://github.com/totilas/muffliato.

**Averaging on synthetic graphs.** We generate synthetic graphs with $n = 2048$ nodes and define the corresponding gossip matrix according to the Hamilton scheme. Note that the

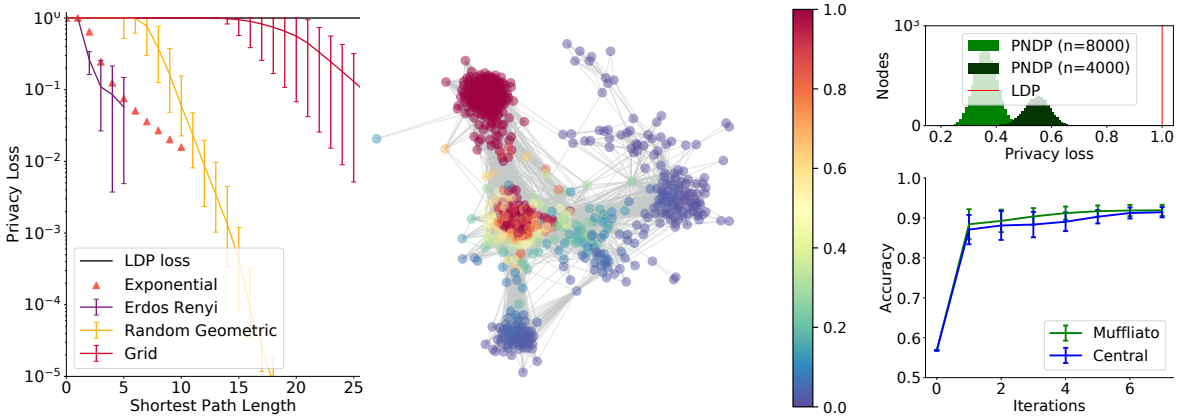



**Figure 6.1** – (a) Left: Privacy loss of *Muffliato* in pairwise NDP on synthetic graphs (best, worst and average in error bars over nodes at a given distance), confirming a significant privacy amplification as the distance increases. (b) Middle: Privacy loss of *Muffliato* from a node chosen at random on a Facebook ego graph, showing that leakage is very limited outside the node's own community. (c) Right: Privacy loss and utility of *Muffliato*-GD when using different Erdös-Rényi graphs after each gradient step, compared to a baseline based on a trusted aggregator.

privacy guarantees of *Muffliato* are deterministic for a fixed $W$, and defined by Equation 6.1. For each graph, we run *Muffliato* for the theoretical number of steps required for convergence, and report in Figure 6.1(a) the pairwise privacy guarantees aggregated by shortest path lengths between nodes, along with the LDP baseline for comparison. *Exponential graph* (generalized hypercube): this has shown to be an efficient topology for decentralized learning [Yin+21]. Consistently with our theoretical result, privacy is significantly amplified. The shortest path completely defines the privacy loss, so there is no variance. *Erdos-Rényi graph* with $q = c \log n/n$ ($c \geq 1$) [ER59], averaged over 5 runs: this has nearly the same utility-privacy trade-off as the exponential graph but with significant variance, which motivates the time-evolving version mentioned in Remark 6.10. *Grid:* given its larger mixing time, it is less desirable than the two previous graphs, emphasizing the need for careful design of the communication graph. *Geometric random graph:* two nodes are connected if and only if their distance is below a given threshold, which models for instance Bluetooth communications (effective only in a certain radius). We sample nodes uniformly at random in the square unit and choose a radius ensuring full connectivity. While the shortest path is a noisy approximation of the privacy loss, the Euclidean distance is a very good estimator as shown in Appendix C.8.

**Averaging on real-world graphs.** We consider the graphs of the Facebook ego dataset [LM12], where nodes are the friends of a given user (this central user is not present is the graph) and edges encode the friendship relation between these nodes. Ego graphs typically induce several clusters corresponding to distinct communities: same high school, same university, same hobbies... For each graph, we extract the giant connected component, choose a user at random and report its privacy loss with respect to other nodes. The privacy loss given by

LDP is only relevant within the cluster of direct neighbors: privacy guarantees with respect to users in other communities are significantly better, as seen in Figure 6.1(b). We observe this consistently across other ego graphs (see Appendix C.8). This is in line with one of our initial motivation: our pairwise guarantees are well suited to situations where nodes want stronger privacy with respect to distant nodes.

**Logistic regression on real-world data.** Logistic regression corresponds to minimizing Equation 6.5 with loss function $\ell(\theta; x, y) = \log(1 + \exp(-y\theta^\top x))$ where $x \in \mathbb{R}^d$ and $y \in \{-1, 1\}$. We use a binarized version of UCI Housing dataset.[3] We standardize the features and normalize each data point $x$ to have unit $L_2$ norm so that the logistic loss is 1-Lipschitz for any $(x, y)$. We split the dataset uniformly at random into a training set (80%) and a test set and further split the training set across users. After each gradient step of *Muffliato*-GD, we draw at random an Erdös-Rényi graph of same parameter $q$ to perform the gossiping step and run the theoretical number of steps required for convergence. For each node, we keep track of the privacy loss towards the first node (note that all nodes play the same role). We report the pairwise privacy loss for this node with respect to all others for $n = 4000$ and $n = 8000$ in Figure 6.1(c) (top). We see that, as discussed in Remark 6.10, time-varying graphs are effective at splitting the privacy loss more uniformly across nodes: the privacy gains over LDP are clear with respect to all nodes. As captured by our theory, these gains increase with the number of nodes $n$ in the system, and they also concentrate better around the mean. We compare the utility of *Muffliato*-GD to a federated learning alternative which uses the same parameters but aggregates noisy model updates using a *trusted* central server rather than by gossiping. As seen in Figure 6.1(c) (bottom), both approaches behave similarly in terms of accuracy across iterations.

## 6.5 Conclusion

We showed that gossip protocols amplify the LDP guarantees provided by local noise injection as values propagate in the graph. Despite the redundancy of gossip that, at first sight, could be seen as an obstacle to privacy, the privacy amplification turns out to be significant: it can nearly match the optimal privacy-utility trade-off of the trusted curator. From the fundamental building block — noise injection followed by gossip — that we analyzed under the name *Muffliato*, one can easily extend the analysis to other decentralized algorithms, such as the dual approach proposed in [Sca+17]. Our results are motivated by the typical relation between proximity in the communication graph and lower privacy expectations. Other promising directions are to assume that closer people are more similar, which leads to smaller individual privacy accounting [FZ21], to design new notions of similarity between nodes in graphs as done in personalization [EMS22] that match the privacy loss variations.

[3] https://www.openml.org/d/823

# Chapter 7

# Random Walks under Network Differential Privacy

In this chapter, we characterize the privacy guarantees of decentralized learning with random walk algorithms, where a model is updated by traveling from one node to another along the edges of a communication graph. We first derive an expression for the complete graph. We then derive closed-form expressions for the privacy loss between each pair of nodes where the impact of the communication topology is captured by graph theoretic quantities. Our results further reveal that random walk algorithms yield better privacy guarantees than gossip algorithms for nodes close to each other. We supplement our theoretical results with an empirical evaluation over synthetic and real-world graphs and datasets.

This chapter corresponds to the last theorem of [CB22] and the paper [CBU24].

## 7.1 Introduction

In the previous chapter, we have seen that gossip algorithms enables good privacy-utility tradeoffs under the Network DP setting. However, a major drawback of gossip algorithms is that, even in their more asynchronous versions, they generate redundant communications and require all nodes to be largely available (since any node can be updated at any time). Redundant communication and availability has been touted as a major obstacle in distributed private learning [STU17]. In this chapter, we study random walk algorithms under the network differential privacy setting defined in Chapter 5. The random walk approach has been previously analyzed for the ring graph (see Section 5.4), which has the advantage of being deterministic and always having the same interval between two nodes participations, easing the privacy guarantees computations. In this chapter, we remove this hypothesis and tackle general network topologies.

We first consider the case of random walks over a complete graph in Section 7.2. We provide an algorithm for real summation and prove a privacy amplification result of $O(1/\sqrt{n})$ compared to the same algorithm analyzed under LDP, again matching the privacy-utility trade-off of the trusted curator model. We also discuss a natural extension for computing discrete histograms. Finally, we turn to the task of optimization with stochastic gradient descent and propose a decentralized SGD algorithm that achieves a privacy amplification of $O(\log n/\sqrt{n})$ in some regimes, nearly matching the utility of *centralized* differentially private SGD [BST14]. Interestingly, the above algorithms can tolerate a constant number of collusions at the cost of some reduction in the privacy amplification effect. At the technical level, our theoretical analysis leverages results on privacy amplification by subsampling [BBG18], shuffling [Erl+19; Bal+19b; FMT20] and iteration [Fel+18] in a novel decentralized context: this is made possible by the restricted view of participants offered by decentralized algorithms and adequately captured by our notion of network DP. We show experimentally in Section 7.2.4 that privacy gains are significant in practice both for simple analytics and for training models in federated learning scenarios.

We then generalize to arbitrary communication graphs, raising even more challenging privacy analysis. In Section 7.3, we first introduce a private version of decentralized *stochastic gradient descent* (SGD) based on random walks on arbitrary graphs: in a nutshell, the node holding the model at a given step updates it with a local SGD step, adds Gaussian noise and forwards it to one of its neighbor chosen with appropriate probability. Focusing on the strongly convex setting, we then establish a convergence rate for our algorithm by building upon recent results on SGD under Markovian sampling [Eve23], and show that the result compares favorably to its gossip SGD counterpart. Our main contribution lies in precisely characterizing the privacy loss between all pairs of nodes using a PNDP analysis. We obtain elegant closed-form expressions that hold for arbitrary graphs, capturing the effect of a particular choice of graph through graph-theoretic quantities. We also show how our general closed-form expression yields explicit and interpretable results for specific graphs. Finally, we use synthetic and real graphs and datasets to illustrate our theoretical results and show their practical relevance compared to the gossip algorithms analyzed in previous work.

In summary, our contributions are as follows:

1. We propose a private version of random walk stochastic gradient descent for arbitrary graphs (Algorithm 7.3);
2. We establish its convergence rate for strongly convex loss functions (Theorem 7.6);
3. We derive closed-form expressions for the privacy loss between each pair of nodes that capture the effect of the topology by graph-theoretic quantities (Theorem 7.7);

**Algorithm 7.1:** Private summation on a complete graph

$\tau \leftarrow 0$, $k_1 \leftarrow 0, \dots, k_n \leftarrow 0$
**for** $t = 1$ **to** $T$ **do**
  Draw $u \sim \mathcal{U}(1, \dots, n)$
  $k_u \leftarrow k_u + 1$
  $\tau \leftarrow \tau + \text{Perturb}(x_u^{k_u}; \sigma_{loc})$
**return** $\tau$

4. We theoretically and experimentally compare our guarantees to those of gossip algorithms, highlighting the superiority of our approach in several regimes.

## 7.2 Walk on complete graph

In this section, we consider the case of a random walk on the complete graph. In other words, at each step, the token is sent to a user chosen uniformly at random among $\mathcal{V}$. We consider random walks of fixed length $T > 0$, hence the number of times a given user contributes is itself random. We assume the token path to be hidden, including the previous sender and the next receiver, so the only knowledge of a user is the content of the messages that she/he receives and sends.

### 7.2.1 Real Summation

For real summation, we consider the simple and natural protocol shown in Algorithm 7.1: a user $u$ receiving the token $\tau$ for the $k$-th time updates it with $\tau \leftarrow \tau + \text{Perturb}(x_u^k; \sigma_{loc})$. As in Section 5.4.1, $\sigma_{loc}$ is set such that $\text{Perturb}(\cdot; \sigma_{loc})$ satisfies $(\varepsilon, \delta)$-LDP, and thus implicitly depends on $\varepsilon$ and $\delta$. We now show network DP guarantees, which rely on *the intermediate aggregations of values between two visits of the token to a given user* and *the secrecy of the path taken by the token*. For clarity, the theorem below gives only the main order of magnitude, but the complete and tighter formula can be found in Appendix D.1.

**Theorem 7.1.** *Let $\varepsilon < 1$ and $\delta > 0$. Algorithm 7.1 outputs an unbiased estimate of the sum of $T$ contributions with standard deviation $\sqrt{T}\sigma_{loc}$, and satisfies $(\varepsilon', (N_v + T/n)\delta_{cycle} + \delta' + \hat{\delta})$-network DP for all $\delta', \hat{\delta} > 0$ with*

$$\varepsilon' = O\Big(\sqrt{N_v \log(1/\delta')}\varepsilon/\sqrt{n}\Big), \tag{7.1}$$

*where* $N_v = \frac{T}{n} + \sqrt{\frac{3}{n} T \log(1/\hat{\delta})}$ *and* $\delta_{cycle} = f(\varepsilon, \delta, n)$ *is given by amplification by subsampling for* $n$ *points with replacement* [BBG18] *in a set of size* $n$.[a] *".*

[a] See Appendix D.1.1 for the precise formula. In practice, we numerically observe that $\delta_{cycle} \leq \delta$ in our experiments.

*Sketch of proof.* We summarize here the main steps (see Appendix D.1 for details). We fix a user $v$ and quantify how much information about the private data of another user $u$ is leaked to $v$ from the visits of the token. The number of visits to $v$ follows a binomial law $\mathcal{B}(T, 1/n)$ that we upper bound by $N_v$ using Chernoff with probability $1 - \hat{\delta}$. Then, for a contribution of $u$ at time $t$, it is sufficient to consider the cycle formed by the random walk between the two successive passages in $v$ containing $t$. To be able to use amplification by subsampling [BBG18], we actually consider a fictive walk where each cycle cannot exceed a length of $n$: if a cycle is larger, we assume that the value of the token is observed by $v$ every $n$ steps. As the information leaked to $v$ by the actual walk can be obtained by post-processing of this fictive walk, it is enough to compute the privacy loss of the fictive walk, which has at most $N_v + T/n$ cycles (with high probability). Then, we prove that each cycle incurs at most a privacy loss of $3\varepsilon/\sqrt{n}$ by combining intermediate aggregations and amplification by subsampling. We conclude with $\varepsilon' = O(\sqrt{(N_v + T/n)\log(1/\delta')}\frac{\varepsilon}{\sqrt{n}})$ and $\delta_f = (N_v + T/n)\delta_{cycle} + \delta' + \hat{\delta}$ by advanced composition. ■

The same algorithm analyzed under LDP yields $\varepsilon' = O(\sqrt{N_v \log(1/\delta')}\varepsilon)$, which is optimal for averaging $N_v$ contributions per user in the local model. For $T = \Omega(n)$, Theorem 7.1 thus shows that network DP asymptotically provides a privacy amplification of $O(1/\sqrt{n})$ over LDP and matches the privacy-utility trade-off of a trusted aggregator. We will see in Section 7.2.4 that our complete (tighter) formula given in Appendix D.1 improves upon local DP as soon as $n \geq 20$ (Figure 7.1(a)). We also show that the gains are significantly stronger in practice than what our theoretical results guarantee (Figure 7.1(b)).

**Extension to discrete histogram computation.** We can obtain a similar result for histograms by bounding the privacy loss incurred for each cycle by using amplification by shuffling [Erl+19; Bal+19b; FMT20], similar to what we did for the ring topology (Section 5.4.2). Details are in Appendix D.2.

### 7.2.2 Optimization with SGD

We now turn to the task of private convex optimization with stochastic gradient descent (SGD). Let $\mathcal{W} \subseteq \mathbb{R}^d$ be a convex set and $f(\cdot; D_1), \ldots, f(\cdot; D_n)$ be a set of convex $L$-Lipschitz and $\beta$-smooth functions over $\mathcal{W}$ associated with each user. We denote by $\Pi_{\mathcal{W}}(w) = \arg\min_{w' \in \mathcal{W}} \|w - w'\|$ the Euclidean projection onto the set $\mathcal{W}$. We aim to privately solve the following optimiza-

**Algorithm 7.2:** Private SGD on a complete graph

1 Initialize $\tau \in \mathcal{W}$
2 **for** $t = 1$ **to** $T$ **do**
3 Draw $u \sim \mathcal{U}(1, \dots, n)$
4 $Z = [Z_1, \dots, Z_d], \quad Z_i \sim \mathcal{N}(0, \frac{8L^2 \log(1.25/\delta)}{\varepsilon^2})$
5 $\tau \leftarrow \Pi_{\mathcal{W}}(\tau - \eta(\nabla_\tau f(\tau; D_u) + Z))$
6 **return** $\tau$

tion problem:

$$w^* \in \arg\min_{w \in \mathcal{W}} \{F(w) := \frac{1}{n} \sum_{u=1}^{n} f(w; D_u)\}. \tag{7.2}$$

Eq. 7.2 encompasses many machine learning tasks (e.g., ridge and logistic regression, SVMs, etc).

To privately approximate $w^*$, we propose Algorithm 7.2. Here, the token $\tau \in \mathcal{W}$ represents the current iterate. At each step, the user $u$ holding the token performs a projected noisy gradient step and sends the updated token to a random user. We rely on the Gaussian mechanism to ensure that the noisy version of the gradient $\nabla_\tau f(\tau; D_u) + Z$ satisfies $(\varepsilon, \delta)$-LDP: the variance $\sigma^2$ of the noise in line 4 of Algorithm 7.2 follows from the fact that gradients of $L$-Lipschitz functions have sensitivity bounded by $2L$ [BST14]. Our network DP guarantee is stated below, again in a simplified asymptotic form.

**Theorem 7.2.** *Let $\varepsilon < 1$, $\delta < 1/2$. Alg. 7.2 with $\eta \leq 2/\beta$ achieves $(\varepsilon', \delta + \hat{\delta})$-network DP for all $\hat{\delta} > 0$ with*

$$\varepsilon' = \sqrt{2q \log(1/\delta)}\varepsilon / \sqrt{\log(1.25/\delta)}, \tag{7.3}$$

*where $q = \max\left(\frac{2N_u \log n}{n}, 2\log(1/\delta)\right)$ and $N_u = \frac{T}{n} + \sqrt{\frac{3}{n} T \log(1/\hat{\delta})}$.*

*Sketch of proof.* The proof tracks the evolution of the privacy loss using Rényi Differential Privacy (RDP) [Mir17] and leverages amplification by iteration [Fel+18] in a novel decentralized context. We give here a brief sketch (see Appendix D.3 for details). Let us fix two users $u$ and $v$ and bound the privacy leakage of $u$ from the point of view of $v$. We again bound the number of contributions $N_u$ of user $u$, but unlike in the proof of Theorem 7.1 we apply this result to the user releasing information (namely $u$). We then compute the network RDP guarantee for a fixed contribution of $u$ at time $t$. Crucially, it is sufficient to consider the first time $v$ receives the token at a step $t' > t$. Privacy amplification by iteration tells us that the larger $t'$, the less is learned by $v$ about the contribution of $u$. Note that $t'$ follows a geometric law of parameter $1/n$. Using the weak convexity of the Rényi divergence [Fel+18], we can bound the Rényi divergence $D_\alpha(Y_v || Y'_v)$ between two random executions $Y_v$ and $Y'_v$ stopping at $v$ and differing only in the contribution of $u$ by the expected divergence over the geometric distribution. Combining with

amplification by iteration eventually gives us $D_\alpha(Y_v||Y'_v) \leq 4\alpha L^2 \log n/\sigma^2 n$. We apply the composition property of RDP over the $N_u$ contributions of $u$ and convert the RDP guarantee into $(\varepsilon, \delta)$-DP. ■

Algorithm 7.2 is also a natural approach to private SGD in the local model, and achieves $\varepsilon' = O(\sqrt{N_u \log(1/\delta')}\varepsilon)$ under LDP. Thus, for $T = \Omega(n^2\sqrt{\log(1/\delta)}/\log n)$ iterations, Theorem 7.2 gives a privacy amplification of $O(\log n/\sqrt{n})$ compared to LDP. Measuring utility as the amount of noise added to the gradients, the privacy-utility trade-off of Algorithm 7.2 in network DP is thus nearly the same (up to a log factor) as that of private SGD in the trusted curator model![1] For smaller $T$, the amplification is much stronger than suggested by the simple closed form in Eq. 7.3: we can numerically find a smaller $\varepsilon'$ that satisfy the conditions required by our non-asymptotic result, see Appendix D.3 for details.

We note that we can easily obtain utility guarantees for Algorithm 7.2 in terms of optimization error. Indeed, the token performs a random walk on a complete graph so the algorithm performs the same steps as a *centralized* (noisy) SGD algorithm. We can for instance rely on a classic result by Shamir and Zhang [SZ13, Theorem 2 therein] which shows that SGD-type algorithms applied to a convex function and bounded convex domain converge in $O(1/\sqrt{T})$ as long as gradients are unbiased with bounded variance.

**Proposition 7.3.** *Let the diameter of $\mathcal{W}$ be bounded by $D$. Let $G^2 = L^2 + \frac{8dL^2\log(1.25/\delta)}{\varepsilon^2}$, and $\tau \in \mathcal{W}$ be the output of Algorithm 7.2 with $\eta = D/G\sqrt{t}$. Then we have:*

$$\mathbb{E}[F(\tau) - F(w^*)] \leq 2DG(2 + \log T)/\sqrt{T}.$$

A consequence of Proposition 7.3 and Theorem 7.2 is that for fixed privacy budget and sufficiently large $T$, the expected error of Algorithm 7.2 is $O(\log n/\sqrt{n})$ smaller under network DP than under LDP.

### 7.2.3 Discussion

An advantage of considering a random walk over a complete graph is that our approach is naturally robust to the presence of a (constant) number of colluding users. Indeed, when $c$ users collude, they can be seen as a unique node in the graph with a transition probability of $\frac{c}{n}$ instead of $\frac{1}{n}$. We can then easily adapt the proofs above, as the total number of visits to colluding users follows $\mathcal{B}(T, c/n)$ and the size of a cycle between two colluding users follows a geometric law of parameter $1 - c/n$. Hence, we obtain the same guarantees under Definition 5.2 as for the case with $n/c$ non-colluding users under Definition 5.1. Interestingly, these privacy

[1] Incidentally, the analysis of centralized private SGD [BST14] also sets the number of iterations to be of order $n^2$.

guarantees hold even if colluding users bias their choice of the next user instead of choosing it uniformly. Indeed, as soon as colluded nodes do not hold the token, the random walk remains unbiased, with the same distribution for the time it takes to return to colluders (i.e., $\mathcal{G}(m/n)$ for $m$ colluded nodes).

We note that despite the use of a complete graph, all users do not necessarily need to be available throughout the process. For instance, if we assume that the availability of a user at each time step follows the same Bernoulli distribution for every user, we can still build a random walk with the desired distribution, similarly to what is done in another context by [Bal+20b].

On the other hand, the assumption that users do not know the identity of the previous sender and the next receiver may seem quite strong. It is however possible to lift this assumption by bounding the number of times that a contribution of a given user $u$ is directly observed by a given user $v$ separately and adding the corresponding privacy loss to our previous results. This additional term dominates the others for small values of $T$ due to its large variance (an "unlucky" node may forward a lot of times the token to the same node). But as the expected number of contributions per node increases, the relative importance of this term decreases (and thus the privacy amplification increases) until $T = \Omega(n^2)$, for which the amplification reaches the same order as in Theorems 7.1-7.2. Even though $T = \Omega(n^2)$ is seldom used in practice, we note that it is also required to obtain optimal privacy amplification by iteration under multiple contributions per user [Fel+18]. Moreover, in practical implementations, we can mitigate the large variance effect in the regime where $T = o(n^2)$ by enforcing a *deterministic* bound on the number of times any edge $(u, v)$ is used, e.g., by contributing only noise along $(u, v)$ after it has been used too many times. We refer to Appendix D.4 for details and formal derivations.

### 7.2.4 Experiments

We now present some numerical experiments that illustrate the practical significance of our privacy amplification results in the complete graph setting (Section 7.2).[2]

#### Real Summation

**Comparison of analytical bounds.** We numerically evaluate the theoretical (non-asymptotic) bound of Theorem 7.1 for the task of real summation and compare it to local DP. Recall that the number of contributions of a user is random (with expected value $T/n$). For a fair comparison between network and local DP, we derive an analogue of Theorem 7.1 for local DP. In addition, to isolate the effect of the number of contributions (which is the same in both settings), we also report the bounds obtained under the assumption that each user contributes exactly $T/n$

[2]The code is available https://github.com/totilas/privacy-amplification-by-decentralization

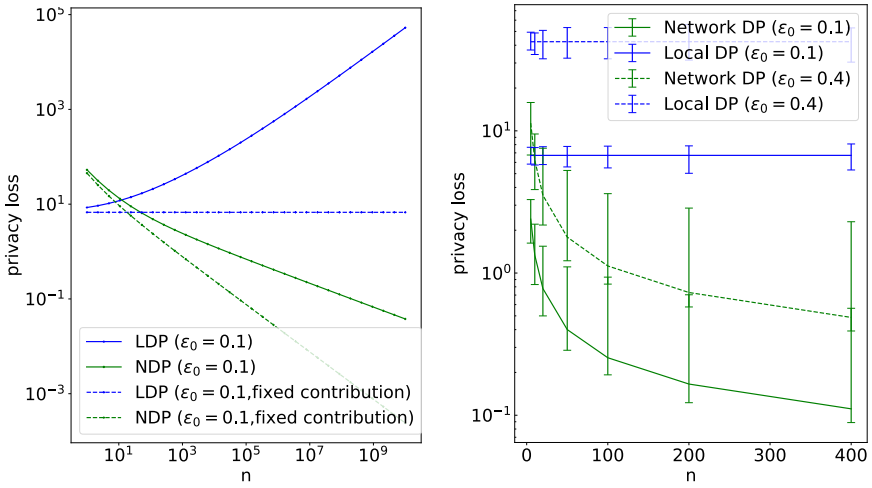



**Figure 7.1** – Comparing network and local DP on real summation for $T = 100n$. $\varepsilon_0$ rules the amount of local noise added to each contribution (i.e., each single contribution taken in isolation satisfies $\varepsilon_0$-LDP). For the empirical results of Figure 7.1(b), the curves report the average privacy loss across all pairs of users and all 10 random runs; error bars give best and worst cases.

times. Figure 7.1(a) plots the value of the bounds for varying $n$. We see that our theoretical result improves upon local DP as soon as $n \geq 20$, and these gains become more significant as $n$ increases. We note that the curves obtained under the fixed number of contributions per user also suggest that a better control of $N_v$ in the analytical bound could make our amplification result significantly tighter.

**Gap with empirical behavior.** Our formal analysis involves controlling the number of contributions of users, as well as the size of cycles using concentration inequalities, which require some approximations. In practical deployments one can instead use the actual values of these quantities to compute the privacy loss. We thus investigate the gap between our theoretical guarantees and what can be obtained in practice through simulations. Specifically, we sample a random walk of size $T = 100n$. Then, for each pair of users, we compute the privacy loss based on the actual walk and the advanced composition mechanism. We repeat this experiment over 10 random walks and we can then report the average, the best and the worst privacy loss observed across all pairs of users and all random runs. Figure 7.1(b) reports such empirical results obtained for the case of real summation with the Gaussian mechanism, where the privacy grows with a factor $\sqrt{m}$ where $m$ is the number of elements aggregated together (i.e., the setting covered by Theorem 7.1). We observe that the gains achieved by network DP are significantly stronger in practice than what our theoretical bound guarantees, and are significant even for small $n$ (see Figure 7.1(a)). Our experiments on discrete histogram computation also show significant gains (see Appendix D.5).

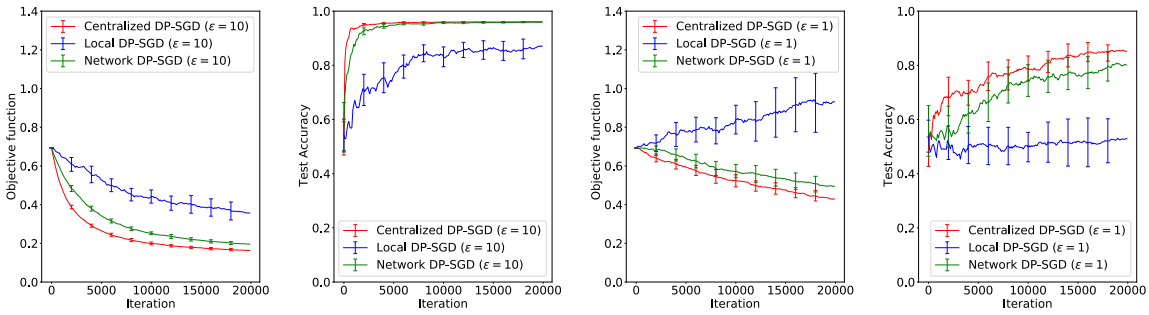



**Figure 7.2** – Comparing three settings for SGD with gradient perturbation. Unlike Local and Network DP-SGD, Centralized DP-SGD requires a trusted curator and benefits from amplification by subsampling. Network DP nearly bridges the gap between Centralized and Local DP-SGD. In all methods, $\sigma$ is set to ensure $\varepsilon = 10$ (left plots) or $\varepsilon = 1$ (right plots), and $\delta = 10^{-6}$. Mean and standard deviations are computed over 20 runs.

### Machine Learning with SGD

We now present some experiments on the task of training a logistic regression model in the decentralized setting. Logistic regression corresponds to solving Eq. 7.2 with $\mathcal{W} = \mathbb{R}^d$ and the loss functions defined as $f(w; D_u) = \frac{1}{|D_u|} \sum_{(x,y)\in D_u} \log(1 + \exp(-yw^\top x))$ where $x \in \mathbb{R}^d$ and $y \in \{-1, 1\}$. We use a binarized version of UCI Housing dataset.[3] We standardize the features and further normalize each data point $x$ to have unit L2 norm so that the logistic loss is 1-Lipschitz for any $(x, y)$. We split the dataset uniformly at random into a training set (80%) and a test set, and further split the training set across $n = 2000$ users, resulting in each user $u$ having a local dataset $D_u$ of size 8.

We compare three variants of private SGD based on gradient perturbation with the Gaussian mechanism. *Centralized DP-SGD* is the centralized version of differentially private SGD introduced by [BST14], which assumes the presence of a trusted curator/aggregator. *Local DP-SGD* corresponds to Algorithm 7.2 with the noise calibrated for the LDP setting. Finally, *Network DP-SGD* is Algorithm 7.2 with the noise calibrated according to network DP (see Theorem 7.2). To make the comparison as fair as possible, all approaches (including Centralized DP-SGD) use the full dataset $D_u$ of a randomly chosen user $u$ as the mini-batch at each step.

Given the privacy budget $(\varepsilon, \delta)$ for the whole procedure, each of the three methods leads to a different choice for $\sigma$ that parametrizes the level of noise added to each gradient. In our experiments, we fix $\varepsilon = 10$ (low privacy) and $\varepsilon = 1$ (stronger privacy) and $\delta = 10^{-6}$. We recall that we consider user-level DP ($X \sim_u X'$ differ in the local database of user $u$). Note that due to composition, more iterations increase the per-iteration level of noise needed to achieve a fixed DP guarantee. As the number of contributions of a given user is random, we upper bound it in advance with a tighter bound than used in our theorems, namely $cT/n$ where $c$

[3] https://www.openml.org/d/823

is a parameter to tune. If a user is asked to participate more times than budgeted, it simply forwards the token to another user without adding any contribution. In the case of Network DP-SGD, the user still adds noise as the privacy guarantees of others rely on it. Note that the best regime for network DP is when the number of contributions of a user is roughly equal to $n$, see Theorem 7.2. In our experiments, we are not in this regime but the privacy amplification effect is stronger than the closed form of the theorem. In practice, we compute numerically the smallest $\sigma$ needed to fulfill the conditions of the proof (see Appendix D.3).

Figure 7.2 shows results for $T = 20000$, where the step size $\eta$ was tuned separately for each approach in $[10^{-4}, 2]$. We see that Network DP-SGD nearly matches the privacy-utility trade-off of Centralized DP-SGD for both $\varepsilon = 1$ and $\varepsilon = 10$ without relying on a trusted curator. Network DP-SGD also clearly outperforms Local DP-SGD, which actually diverges for $\varepsilon = 1$. These empirical results are consistent with our theory and show that Network DP-SGD significantly amplifies privacy compared to local DP-SGD even when the number of iterations $T$ is much smaller than $O(n^2/\log n)$, a regime which is of much practical importance.

## 7.3 Extension to arbitrary graphs

In the rest of the chapter, we study random walk SGD, as defined in Section 2.4, incorporating Gaussian noise injection, as previously done for the complete graph and recapped in Algorithm 7.3. To align with the published version [CBU24], we remove the assumption that the gossip matrix is doubly stochastic and allow for any stationary distribution solely for optimization analysis. In this context, we examine the following minimization problem.

$$f(x) = \sum_{v=1}^{n} \pi_v f_v(x), \tag{7.4}$$

where $x \in \mathbb{R}^d$ represents the parameters of the model and the local function $f_v$ depends only on the local dataset of node $v$, and $\pi_v \geq 0$ is the weight given to $f_v$ (in practice, the vector $\pi$ will correspond to the stationary distribution of the random walk as defined below).

For the random walk algorithms we will consider, the complete output $\mathcal{A}(\mathcal{D})$ consists of the trajectory of the token and its successive values during training. At a given step, the token of the random walk shares its current value only with its current location, but the other nodes cannot see this state. Thus, we define the view of a node $v$ as

$$\begin{aligned}\mathcal{O}_v(\mathcal{A}(\mathcal{D})) = \{(t, x_t, w) :\ &\text{the token } x_t \text{ was in } v \\ &\text{at time } t \text{ and then sent to } w\}\,.\end{aligned} \tag{7.5}$$

**Algorithm 7.3:** PRIVATE RANDOM WALK GRADIENT DESCENT (RW DP-SGD)

**Input:** transition matrix $W$ on a graph $G$, number of iterations $T$, noise variance $\sigma^2$, starting node $v_0$, initial token value $x_0$, step size $\gamma$, gradient sensitivity $\Delta$, local loss function $f_v$
**for** $t = 0$ *to* $T-1$ **do**
  Draw $\eta \sim \mathcal{N}(0, \Delta^2\sigma^2)$
  Compute $g_t$ s.t. $\mathbb{E}[g_t] = \nabla f_{v_t}(x_t)$
  $x_{t+1} \leftarrow x_t - \gamma(g_t + \eta)$
  Draw $u \sim W_{v_t}$ in the set of neighbors of $v_t$
  Send token to $u$
  $v_{t+1} \leftarrow u$

In this definition, nodes know to whom they send the token, but not from whom they receive it. Ensuring the anonymity of the sender can be achieved by using mix networks [SP06] or anonymous routing [DMS04]. However, our results directly extend to the case where the sender's anonymity cannot be ensured, see Remark 7.9 in Section 7.3.1.

### Private SGD with Random Walks

In this section, we introduce a decentralized *stochastic gradient descent* (SGD) random walk algorithm to privately approximate the minimizer of (7.4), and analyze its convergence in the strongly convex case. This algorithm, presented in Algorithm 7.3, generalizes the private random walk algorithm on the complete graph, introduced and analyzed by Cyffers and Bellet [CB22], to arbitrary graphs. Differential privacy is achieved by adding Gaussian noise to the local gradient at each step. The step size is constant over time as commonly done in (centralized) differentially private stochastic gradient descent (DP-SGD) [BST14].

The non-private version of this algorithm converges in various settings (see Section 2.4). In this chapter, we adapt a recent proof for the non-private version [Eve23]. For simplicity, we focus on strongly convex and smooth objectives with bounded gradients at the global optimum.

**Assumption 7.4** (Bounded gradient and strong convexity)**.** *We assume that $f$ is $\mu$-strongly convex and $L$-smooth. Let $x^*$ be its minimizer. We assume that, for $\zeta_* \geq 0$, $\forall v \in V, \|\nabla f_v(x^*)\|^2 \leq \zeta_*^2$.*

In the case of stochastic gradient descent, stochasticity also comes from the fact that we sample from the local dataset. To handle both cases, we define $g_t$ as an unbiased estimator of $\nabla f_{v_t}(x_t)$. We thus require to bound the variance of this estimator.

**Assumption 7.5** (Bounded local noise)**.** *We assume that the stochastic gradients respect the following condition:* $\mathbb{E}\left[\left\|g_t - \nabla f_{v_t}\left(x_t\right)\right\|^2 \mid x_t, v_t\right] \leqslant \sigma_{sgd}^2$.

**Theorem 7.6.** *Under Assumptions 7.4, and 7.5, for step size* $\gamma = \min\left(\frac{1}{L}, \frac{1}{T\mu}\log\left(T\frac{\|x_0-x^\star\|^2}{\frac{39L}{\mu^2}\tau_{\text{mix}}\zeta_\star^2}\right)\right)$ *the iterates verify:*

$$\mathbb{E}(\|x_T - x^*\|^2) \leq 2e^{-\frac{T\mu}{L}}\|x_0 - x^*\|^2 + \left(\frac{39\tau_{\text{mix}}\zeta_*^2 L}{\mu^3 T} + \frac{(d\sigma^2\Delta^2 + \sigma_{sgd}^2)L}{\mu^2 T}\right)\log\frac{T\mu^2\|x_0 - x^*\|^2}{39L\tau_{\text{mix}}\zeta_*^2}.$$

*Proof.* We adapt the proof from Even [Eve23] that keeps track of the shift due to the Markov sampling of the random walk thanks to a comparison with delayed gradient. Following the same bounding steps and taking expectation over the noise distribution we obtain a similar inequality on the iterates than the non-private version up to an additional term for the noise. Setting the step size to balance the terms leads to the final equality. See Appendix D.6 for a full proof. ■

The convergence rate in Theorem 7.6 has three terms: the exponential convergence towards the minimizer parameterized by the condition number $L/\mu$ of $f$, the impact of the stochasticity of the random walk in $\tilde{\mathcal{O}}(\tau_{\text{mix}}\zeta_*^2 L/\mu^3 T)$ and the additional term due to the noise injection in $\tilde{\mathcal{O}}(\sigma^2\Delta^2 L/\mu^2 T)$. A similar term would appear when adapting the non-private convergence proof for other settings such as under the Polyak-Lojasiewicz condition.

**Comparison with private gossip [Cyf+22].** In our random walk algorithm, each step involves computing the gradient of a single node and a single message, whereas the private gossip SGD algorithm of Cyffers et al. [Cyf+22] alternates between the computation of local gradients at all nodes in parallel and a multi-step gossip communication phase until (approximate) consensus. Rephrasing the result of Cyffers et al. [Cyf+22] in our notation, each communication phase in Cyffers et al. [Cyf+22] requires $\tilde{\mathcal{O}}(\tau_{\text{mix}} \cdot \log(\zeta_*^2/\sigma^2))$ steps where all the nodes send updates synchronously. For the optimization part, the first term is the same in $\mathcal{O}(e^{-\frac{T\mu}{L}}\|x_0 - x^*\|^2)$, but there is only one other term in $\mathcal{O}(\sigma^2 L/n\mu T)$. Hence, we lose a factor of $n$, but the reduced communication compensates for this. Our analysis is tighter in the sense that we can separate the privacy noise that is independent of the Markov chain, thereby improving the rate of the spectral gap factor compared to a naive analysis. In contrast, the private gossip analysis casts this noise as gradient heterogeneity, because it re-uses the non-private convergence analysis of Koloskova et al. [Kol+20].

**Special case of averaging.** One can use the above algorithm to privately compute the average of values at each node. In this case, we assume that each node has a private value $y_v$ (a float or a vector) and define the local objective function as:

$$f_v(x) = \|x - y_v\|^2. \tag{7.6}$$

Note that, in this case, we have $L = \mu = 2$. A natural approach is to compute the running average of the visited values with noise injection at each step, which corresponds to Algorithm 7.3 with a decreasing step-size $\gamma_t = 1/t$. The drawback of this approach is that damping the first terms of the sum (when the Markov chain is not yet well mixed) requires a lot of steps and results in slow convergence (at least $\tau_{\text{mix}}$ steps). Adopting a constant step-size instead, as in Algorithm 7.3, does not modify the convergence leading terms that stay in $\mathcal{O}(1/T)$ for an adequate step-size.

One way to completely remove the influence of the first terms is to have a *burn-in* phase, when the token walks without performing any update, to come closer to the stationary distribution. Then, after $\tau_{\text{mix}}(\iota/2)$ steps, a running average of $4\delta^2/(\iota^3\lambda_W)$ is obtained, as proven in Theorem 12.21 of [LP17].

### 7.3.1 Privacy Analysis

In this section, we derive the privacy guarantees of Algorithm 7.3 for arbitrary graphs and show how this leads to improved trade-offs for specific graphs widely used in decentralized learning. Our main result is a closed-form expression for the privacy loss between each pair of nodes, which holds for arbitrary graphs.

**Theorem 7.7.** *Consider a graph $G$ with transition matrix $W$. After $T$ iterations, for a level of noise $\sigma^2 \geq 2\alpha(\alpha-1)$, the privacy loss of Algorithm 7.3 from node $u$ to $v$ is bounded by:*

$$\varepsilon_{u\to v} \leq \mathcal{O}\left(\frac{\alpha T \log(T)}{\sigma^2 n^2} - \frac{\alpha T}{\sigma^2 n}\log\left(I - W + \frac{1}{n}\mathbb{1}\mathbb{1}^\top\right)_{uv}\right).$$

We recall that the logarithm of a matrix corresponds to the matrix whose eigenvalues are the log of the original eigenvalues and the eigenvectors remain identical. In particular, if $\lambda_1, \cdots, \lambda_n$ are $n$-eigenvalues (counting multiplicity) of a bistochastic symmetric matrix $M$ and $x_1, \cdots, x_n$ are the corresponding eigenvectors, then $\log(M) = \sum_{i=1}^n x_i x_i^\top \log(\lambda_i)$. Note that the fact that $W$ is symmetric ensures this decomposition exists.

*Sketch of proof.* We give a high-level overview of the proof here and refer to Appendix D.7 for details. We fix the two vertices $u$ and $v$ and see how a token visit to $u$ will leak information to $v$. By the post-processing property of RDP, it is sufficient to compute the privacy loss that occurs when the token reaches for the first time $v$ after the visit in $u$. For computing this loss, we use the weak convexity of the Rényi divergence [Fel+18] to condition over the number of steps before reaching $v$. The length of the walk is parameterized by the power of the transition matrix. For a given length, we bound the privacy loss by using privacy amplification by iteration (Theorem 3.11). Refactoring the sum leads to the logarithm of the matrix. We finish by using composition over the $\mathcal{O}(T/n)$ times the token visits $u$ during the walk. ■

**Remark 7.8.** *For the complete graph, the second term is equal to zero as the transition matrix is exactly $W = \frac{1}{n}\mathbb{1}\mathbb{1}^\top$. Thus, we recover the bound of Theorem 7.2. In other words, Theorem 7.7 is a generalization of this previous result to arbitrary graphs.*

**Remark 7.9.** *Theorem 7.7 holds for the definition of the view of the node given in (7.5), where the sender is kept anonymous. We provide in Appendix D.7.1 a similar theorem if the senders are known. In this case, although the formula is more complex, the asymptotic is the same as it roughly shifts the privacy guarantees from one hop.*

### Interpretation of the Formula via Communicability

The privacy loss of Theorem 7.7 has two terms that we can interpret as follows. The first term is the same as in Theorem 7.2 for the complete graph: we have an $O(1/n^2)$ privacy amplification factor compared to local DP, matching what would be obtained in central DP with $n$ users. Our analysis reveals that this term also appears for arbitrary graphs: we can interpret it as a baseline privacy loss that occurs from the collaboration of all agents.

However, in graphs differing from the complete graph, this baseline privacy loss is corrected by the second term which depends on the specific pair $(u, v)$ of the nodes considered. Note that this second term can be negative for some pairs as eigenvector components can be of arbitrary signs. This quantity can be seen as a variant of known graph centrality metrics and, more precisely, communicability.

**Definition 7.10** (Communicability, [EH08; EK15])**.** *For a transition matrix $W$ and $c_i$ a non-increasing positive series ensuring convergence, the communicability between two vertices $u, v$ is defined by:*

$$G_{uv} = \sum_{i=1}^{\infty} c_i W^i_{uv} \,.$$

For $c_i = 1/i!$, this corresponds to the original notion of communicability as presented in [EH08], while for $c_i = a^i$ we recover the Katz centrality [Kat53]. Our formula corresponds to the case $c_i = \frac{\alpha}{\sigma^{2i}}$ as proven in Appendix D.7, and the convergence of the infinite sum is ensured by the fact that we remove the graph component associated with the eigenvalue 1. Note that the coefficient depends on the privacy parameters $\alpha$ and $\sigma$, which might be surprising. Katz centrality does not prescribe a specific value for $a$, except to be small enough – for example, `Networkx` implementation uses the default value $a = 0.1$ –, as the result and ranking tends not to be too sensitive to the choice of $a$.

Communicability is used to detect local structures in complex networks, with applications to community detection and graph clustering, for instance, in physical applications [EK15]. Good

communicability is supposed to capture how well connected are the two nodes in the networks, i.e. how close they are. Hence, having the second term of the privacy loss proportional to the communicability of the nodes shows that our formula matches the intuition that nodes leak more information to closer nodes than to more distant ones.

### 7.3.2 Application to Specific Graphs

We now give closed-form expressions of the privacy loss (7.7) for specific graphs. To show the power of our bound, we note that the results for two network graphs, namely the complete graph in Theorem 7.2 and the ring graph in Theorem 5.5, follow as a corollary of our general result. We illustrate below the possibility to derive closed formulas for other classes of graphs. Proofs are given in Appendix D.10.

**Star graph.** We consider a star graph with a central node in the first position linked to the $n-1$ other nodes. We choose the transition matrix such that the probability of self-loop $\kappa > 0$ is an arbitrarily small constant, and the distribution over the non-central nodes is uniform. Then we have the following privacy guarantees.

**Theorem 7.11.** *Let $u, v \in V$ be two distinct nodes of the star graph and $\kappa > 0$ be an arbitrarily small constant. For a single contribution of node $u$ in Algorithm 7.3 on the star graph, the privacy loss to node $v$ is bounded by:*

$$\varepsilon_{u \to v} \leq \begin{cases} -\frac{\alpha(1-\kappa)}{\sigma^2(n-1)} \log\left(1 - \frac{1}{n-1}\right) & u \neq 1 \text{ and } v \neq 1 \\ \frac{\alpha(1-\kappa)}{2\sigma^2\sqrt{n-1}} \log\left(\frac{\sqrt{n-1}+1}{\sqrt{n-1}-1}\right) & u = 1 \text{ or } v = 1 \end{cases}.$$

*Sketch of proof.* At a high level, as $\kappa$ is an arbitrarily small constant, we can have an upper bound on the entries of all the powers of the adjacency matrix of the star graph. ■

In particular, composing over the $\mathcal{O}(T/n)$ contributions, we see that extremal nodes enjoy a privacy amplification factor of order $\mathcal{O}(n^2)$ and the central node of order $\mathcal{O}(n)$.

**Ring graph.** We consider a symmetric ring where nodes are enumerated from 1 to $n$, which thus slightly differs from the case studied in Yakimenka et al. [Yak+22] and Cyffers and Bellet [CB22], where the ring is directed and thus deterministic [up to the possibility of skipping in Yak+22]. For this graph, our results shows that the amplification is parameterized by the distance between the nodes in the ring.

**Theorem 7.12.** *Let $u, v \in V$ be two distinct nodes of the ring with $a = (u + v - 2) \mod n$ and $\alpha' = \alpha \mathbf{1}_{|u-v|=1}$. For a single contribution of node $u$ in Algorithm 7.3 on the ring graph, the privacy*

*loss to node $v$, $\varepsilon_{u\to v}$, is bounded by:*

$$\frac{\alpha' \log(T) \cos\left(\frac{2\pi a}{n}\right)}{n\sigma^2} + \frac{2\alpha}{n\sigma^2} \sum_{k=1}^{n-1} \cos\frac{\pi a k}{n} \log \frac{3\csc^2(\pi k/n)}{4},$$

*in the case of equal probability between going left, right, or self-looping. Furthermore, if the probability of self-looping is set to $\kappa > 0$, then $\varepsilon_{u\to v}$ is bounded by*

$$\frac{\alpha'}{n\sigma^2} + \frac{\alpha(1-\kappa)}{n\sigma^2} \sum_{t=2}^{T} \sum_{k=1}^{n} \cos^{t-1}\frac{2\pi k}{n} \cos\frac{2\pi(a+1)k}{n}.$$

*Sketch of proof.* At a high level, we use the fact that the adjacency matrix for a ring graph is a circulant matrix, so its eigenvectors are the Fourier modes. Therefore, its full spectral decomposition can be easily computed. ■

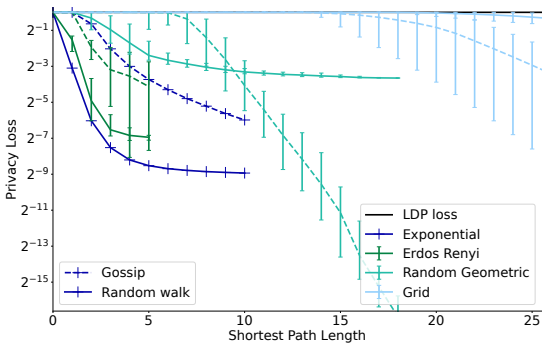
To get an intuition regarding Theorem 7.12, consider two nodes that are close to each other. For simplicity of calculation, consider nodes 1 and 2 and the statement of the theorem. Then $a = 0$ and $\cos^{t-1}(2\pi k/n)\cos(2\pi(a+1)k/n) = \cos^t(2\pi k/n)$.

Therefore, $\varepsilon_{u\to v} \leq \frac{\alpha}{\sigma^2 n} + \frac{\alpha(1-\kappa)}{\sigma^2 n} \sum_{k=1}^{n} \frac{\cos^2(2\pi k/n)}{1-\cos(2\pi k/n)}$.

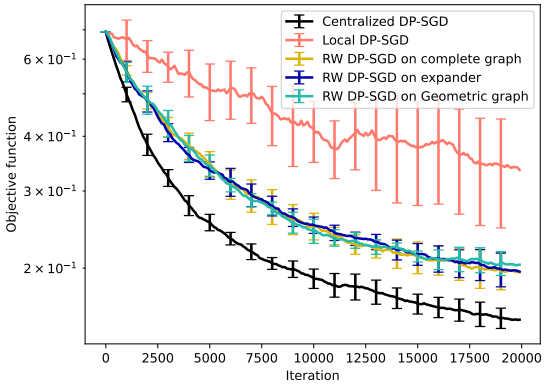
### 7.3.3 Experiments

In this section, we illustrate our results numerically on synthetic and real graphs and datasets and show that our random walk approach achieves superior privacy-utility trade-offs compared to gossip as long as the mixing time of the graph is good enough. The code is available at https://github.com/totilas/DPrandomwalk

**Privacy losses and comparison with the gossip counterpart.** We generate synthetic graphs with $n = 2048$ nodes and report the privacy loss averaged over 5 runs for every pair of nodes of the graphs as a function of the length of their shortest path in Figure 6.1(a). The transition matrix is computed using the *Hamilton weighting*. To compare with the private gossip algorithm [Cyf+22], we consider the task of averaging (thus with $L = \mu = 2$) and consider the same precision level and the same graphs: an exponential graph, an Erdös-Rényi graph with $q = c\log(n)/n$ for $c > 1$, a grid and a geometric random graph. Our private random walk approach incurs a smaller privacy loss for close enough nodes than the private gossip algorithm. Remarkably, our approach improves upon the baseline local DP loss even for very close nodes. Conversely, the privacy loss of our approach is generally higher for more distant nodes. Nevertheless, random walks offer uniformly better privacy guarantees than gossip algorithms for graphs with good connectivity such as Erdös-Rényi graphs or expanders.




(a) Comparison of privacy loss for random walks in bold lines and gossip in dashed lines for the same synthetic graphs with $n = 2048$. Random walks allow privacy amplification even for very close nodes, but the decay is slower than for gossip.




(b) Private logistic regression on the Houses dataset where we compare our RW DP-SGD to with Local and Centralized DP-SGD as baselines.

**Figure 7.3** – Comparison with Muffliato and experiments on Houses

**Logistic regression on synthetic graphs.** We train a logistic regression model on a binarized version of the UCI Housing dataset.[4] The objective function corresponds to $f_v(x) = \frac{1}{|\mathcal{D}_v|}\sum_{(d,y)\in\mathcal{D}_v} \log(1+\exp(-yx^\top d))$ where $d \in \mathbb{R}^d$ and $y \in \{-1,1\}$. As in other chapters, we standardize the features, normalize each data point, and split the dataset uniformly at random into a training set (80%) and a test set (20%). We further split the training set across $2048$ users, resulting in local datasets of $8$ samples each.

In a first experiment, we compare centralized DP-SGD, local DP-SGD, and our random walk-based DP-SGD. For all algorithms, we follow common practice and clip the updates to control the sensitivity tightly. We set $\varepsilon = 1$ and $\delta = 10^{-6}$. Following Chapter 5, we use the mean privacy loss over all pairs of nodes (computed by applying Theorem 7.7) to set the noise level needed for our random walk-based DP-SGD. Figure 7.3(b) reports the objective function through iterations for complete, hypercube, and random geometric graphs. Experimentally, the behavior is the same for all the graphs, meaning that the token walk is diverse enough in every case to have a behavior similar to a uniformly random choice of nodes. The improvement in the privacy-utility trade-off compared to the local DP is significant.

We then compare our random walk algorithm to its gossip counterpart [Cyf+22] on the same logistic regression task. For both algorithms, we fix the mean privacy loss $\bar{\varepsilon}$ across all pairs of nodes to three different levels ($\bar{\varepsilon} \in \{0.5, 1, 2\}$) and we report the accuracy reached by each algorithm on $4$ graphs: complete, exponential, geometric and grid. As shown in Table 7.1, our random walk algorithm outperforms gossip in all cases, which can be explained by a combination of two factors. First, as seen previously in Figure 7.3(a), our algorithm yields a

[4] https://www.openml.org/d/823/

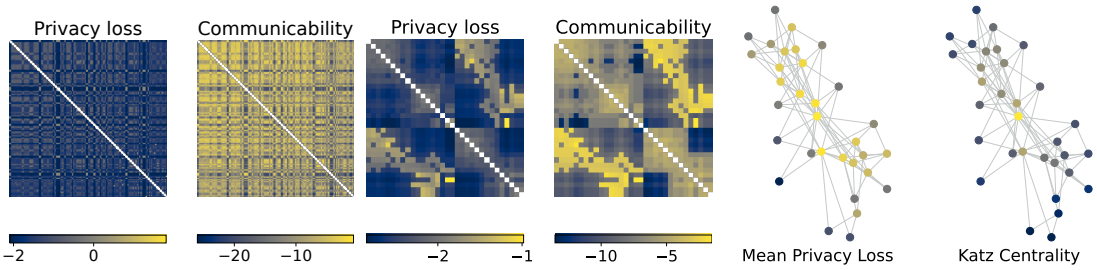
**Table 7.1** – Model accuracy at various mean privacy loss levels averaged over 8 runs.

| Graph | Gossip | Random walk |
|---|---|---|
| **Mean Privacy Loss of 0.5** | | |
| Complete | $0.65 \pm 0.10$ | $0.841 \pm 0.07$ |
| Exponential | $0.70 \pm 0.10$ | $0.818 \pm 0.09$ |
| Geometric | $0.60 \pm 0.07$ | $0.795 \pm 0.06$ |
| Grid | $0.60 \pm 0.07$ | $0.803 \pm 0.10$ |
| **Mean Privacy Loss of 1** | | |
| Complete | $0.70 \pm 0.10$ | $0.900 \pm 0.04$ |
| Exponential | $0.77 \pm 0.05$ | $0.883 \pm 0.05$ |
| Geometric | $0.66 \pm 0.10$ | $0.873 \pm 0.05$ |
| Grid | $0.73 \pm 0.10$ | $0.848 \pm 0.07$ |
| **Mean Privacy Loss of 2** | | |
| Complete | $0.83 \pm 0.06$ | $0.940 \pm 0.02$ |
| Exponential | $0.89 \pm 0.04$ | $0.937 \pm 0.01$ |
| Geometric | $0.67 \pm 0.04$ | $0.933 \pm 0.02$ |
| Grid | $0.72 \pm 0.10$ | $0.919 \pm 0.02$ |

lower mean privacy loss for graphs with good expansion property, especially when the degree of the nodes is high (because the privacy guarantees of gossip degrade linearly with the degree, while random walk is insensitive to it). This gain directly leads to less noise injection when fixing the mean privacy loss, and thus better utility. The second factor that explains why the gain in utility is so pronounced (even for the grid) comes from differences in the SGD version of gossip and random walk. In both methods, the privacy guarantee degrades as a function of the number of participations of nodes (linearly in Rényi DP). In gossip, allowing each node to participate 10 times means that the model will essentially be learned by applying 10 "global" gradient updates (i.e., aggregated over the $n$ nodes), because all local gradients are gossiped until convergence between each gradient computation [see Algorithm 3 in Cyf+22]. In our random walk algorithm, the model is learned by applying $10 \times n$ "local" gradient updates. Even if this represents the same amount of information about the data, better progress is made with many noisy steps than with a small number of less noisy steps, in the same way as mini-batch SGD tends to progress faster than GD in practice.

**Privacy loss on real graphs and communicability.** We consider two real-world graph datasets well-suited for community detection: (i) The Facebook Ego dataset [LM12] represents subgraphs of the Facebook network, where users are nodes and edges correspond to the friendship relation, and each subgraph corresponds to the set of friends of a unique user that is removed from the subgraph; (ii) The Davis Southern women social network [Rob00] is a graph with 32 nodes which corresponds to a bipartite graph of social event attendance by women and has been used in Koloskova, Stich, and Jaggi [KSJ19] and Pasquini, Raynal, and

**Figure 7.4** – Link between graph structure and privacy loss. Left: (a) example of Facebook Ego graph communicability and privacy loss, logarithmic scale. Middle: (b) same on the Southern women graph. Right: (c) the corresponding mean privacy loss and Katz centrality.




Troncoso [PRT23]. We report side-by-side the matrix of pairwise privacy losses and of the communicability in Figure 7.5(a) and Figure 7.5(b). This confirms the similarity between the two quantities as discussed in Section 7.3.1. We also report the Katz centrality compared to the mean privacy loss for each node in Figure 7.5(c), showing that the two quantities also have similar behavior.

We provide other numerical experiments in Appendix D.8: we report the privacy loss on the other Facebook Ego graphs and study the impact of data heterogeneity.

## 7.4 Conclusion

In this chapter, we analyzed the convergence and privacy guarantees of private random walks on complete graphs and then extend the approach to arbitrary graphs. Our results show that random walk-based decentralized algorithms provide favorable privacy guarantees compared to gossip algorithms as presented in Chapter 5, and establish a link between the privacy loss between two nodes and the notion of communicability in graph analysis. Remarkably, as long as the spectral gap of the communication graph is large enough, the random walk approach nearly bridges the gap between the local and the central models of differential privacy. Our results could be broadened by showing convergence under more general hypotheses. Other extensions could include skipping some nodes in the walk, considering latency, or having several tokens running in parallel. Another possible direction is to consider other updates than stochastic gradient descent, such as ADMM algorithms, as done in the next Chapter 8.

# Chapter 8

# From Noisy Fixed-Point Iterations to Private ADMM

In this chapter, we study differentially private machine learning algorithms as instances of noisy fixed-point iterations, in order to derive privacy and utility results from this well-studied framework. We show that this new perspective recovers popular private gradient-based methods like DP-SGD and provides a principled way to design and analyze new private optimization algorithms in a flexible manner. Focusing on the widely-used Alternating Directions Method of Multipliers (ADMM), we use our general framework to derive novel private ADMM algorithms for centralized, federated and decentralized learning. For these three algorithms, we establish strong privacy guarantees leveraging privacy amplification by iteration and by subsampling. Finally, we provide utility guarantees using a unified analysis that exploits a recent linear convergence result for noisy fixed-point iterations.

This chapter corresponds to the paper [CBB23].

## 8.1 Introduction

In this chapter, we take a step back from decentralized algorithms and revisit the general problem of private Empirical Risk Minimization (ERM) from the perspective of *fixed-point iterations* [BC11], which compute fixed points of a function by iteratively applying a non-expansive operator $T$. Fixed point iterations are well-studied and widely applied in mathematical optimization, automatic control, and signal processing. They provide a unifying framework that encompasses many optimization algorithms, from (proximal) gradient descent algorithms to the Alternating Direction Method of Multipliers (ADMM), and come with a rich theory [CP21]. Specifically, we study a general *noisy* fixed-point iteration, where Gaussian noise is added to the operator $T$ at each step. We also consider a (possibly randomized) block-coordinate version,

where the operator is applied only to a subset of coordinates. As particular cases of our framework, we show that we can recover DP-SGD (see Section 3.3.2) and a recent coordinate-wise variant [Man+22]. We then prove a utility bound for the iterates of our general framework by exploiting recent linear convergence results from the fixed-point literature [CP19].

With this general framework and results in place, we show that we can design and analyze new private algorithms for ERM in a principled manner. We focus on ADMM-type algorithms, which are known for their effectiveness in centralized and decentralized machine learning [Boy+11; WO12; WO13; Shi+14; VBT17; TSB22; ZL22]. Based on a reformulation of ERM as a consensus problem, we derive private ADMM algorithms for centralized, federated, and decentralized learning. In contrast to previously proposed private ADMM algorithms that require *ad-hoc* algorithmic modifications and customized theoretical analysis [Hua+19; ZZ17; ZKL18; Din+20], our algorithms and utility guarantees follow directly from our analysis of the general noisy fixed-point iteration. In particular, we are the first to our knowledge to derive a general convergence rate analysis of private ADMM that can be used for the centralized, federated, and decentralized settings. We leverage privacy amplification by iteration [Fel+18] and by subsampling [MTZ19] to prove DP guarantees for our private ADMM algorithms. It extends the use of privacy amplification by iteration technique from DP-SGD to private ADMM algorithms.

More generally, *our work stands out as we are not aware of any prior work that considers the general perspective of noisy-fixed point iterations to design and analyze differentially private optimization algorithms.*

**Related work on Private ADMM.** Due to the flexibility and effectiveness of ADMM for centralized and decentralized machine learning [Boy+11; WO12; Shi+14; VBT17], differentially private versions of ADMM have been studied for the centralized [Sha+21], federated [Hua+19; Cao+21; RK22; Hu+19], and fully decentralized [ZZ17; ZKL18; Din+20] settings. These approaches are based on ad-hoc algorithmic modifications and customized convergence analysis. In particular, the privacy guarantees usually rely on perturbation of the primal variables, often through noise addition to the first-order approximation of the corresponding subproblem, similarly to DP-SGD [ZZ17; Cao+21; Din+20; Sha+21]. This leads to complex convergence analysis with potentially restrictive assumptions, and results in privacy guarantees that are difficult to interpret and are often limited to LDP. In contrast, our framework leads to the addition of noise to the dual variable and the privacy analysis requires no additional hypothesis compared to standard ADMM. Furthermore, our centralized, federated and fully decentralized algorithms and their analysis all naturally follow from our generic (block-wise) noisy fixed point iteration formulation. Lastly, except for Cao et al. [Cao+21] who considered only the trusted server setting, we are the first to achieve user-level DP for federated and fully decentralized ADMM.

Finally, as discussed below, we are the first to show that ADMM can benefit from privacy amplification to obtain better privacy-utility trade-offs.

## 8.2 Background on Fixed-Point Iterations and ADMM

In this section, we introduce the necessary background that will constitute the basis of our contributions. We start by providing basic intuitions and results about the fixed-point iterations framework. Then, we show how ADMM fits into this framework.

### 8.2.1 Fixed-Point Iterations

Let us consider the problem of finding a minimizer (or generally, a stationary point) of a function $f : \mathcal{U} \to \mathbb{R}$, where $\mathcal{U} \subseteq \mathbb{R}^p$. This problem reduces to finding a point $u^* \in \mathcal{U}$ such that $0 \in \partial f(u^*)$, or $\nabla f(u^*) = 0$, when $f$ is differentiable. A generic approach to compute $u^*$ is to iteratively apply an operator $T : \mathcal{U} \to \mathcal{U}$ such that the fixed points of $T$, i.e., the points $u^*$ satisfying $T(u^*) = u^*$, coincide with the stationary points of $f$. The iterative application of $T$ starting from an initial point $u_0 \in \mathcal{U}$ constitutes the *fixed-point iteration* framework [BC11]:

$$u_{k+1} \triangleq T(u_k). \tag{8.1}$$

We denote by $I$ the identity operator, i.e. $I(u) \triangleq u$. To analyze the convergence of the sequence of iterates to a fixed point of $T$, various assumptions on $T$ are considered.

**Definition 8.1** (Non-expansive, contractive, and $\lambda$-averaged operators)**.** *Let $T : \mathcal{U} \to \mathcal{U}$ and $\lambda \in (0, 1)$. We say that:*

- *$T$ is* non-expansive *if it is 1-Lipschitz, i.e., $\|T(u) - T(u')\| \leq \|u - u'\|$ for all $u, u' \in \mathcal{U}$.*
- *$T$ is $\tau$-*contractive *if it $\tau$-Lipschitz with $\tau < 1$.*
- *$T$ is $\lambda$-*averaged *if there exists a non-expansive operator $R$ such that $T = \lambda R + (1 - \lambda)I$.*

Hereafter, we will focus on $\lambda$-averaged operators that correspond to a barycenter between the identity mapping and a non-expansive operator. This family encompasses many popular optimization algorithms. For instance, when $f$ is convex and $\beta$-smooth, the operator $T = I - \gamma\nabla f$, which corresponds to gradient descent, is $\gamma\beta/2$-averaged for $\gamma \in (0, 2/\beta)$. The proximal point, proximal gradient and ADMM algorithms also belong to this family [BC11]. By the Krasnosel'skii Mann theorem [Byr03], the iterates of a $\lambda$-averaged operator converge. Hence, *formulating an optimization algorithm as the application of a $\lambda$-averaged operator allows us to reuse generic convergence results.*

The rich convergence theory of fixed point iterations goes well beyond the simple iteration (8.1), see [CP21] for a recent overview. In this chapter, we leverage several extensions of this theory. First, we consider *inexact updates*, where each application of $T$ is perturbed by additive noise of bounded magnitude. Such noise can arise because the operator is computed only approximately (for higher efficiency) or due to the stochasticity in data-dependent computations. Another extension considers $T$ operating on a decomposable space $\mathcal{U} = \mathcal{U}_1 \times \cdots \times \mathcal{U}_B$ with $B$ blocks, i.e.,

$$T(u) \triangleq (T_1(u), \ldots, T_B(u)), \quad \text{where } T_b : \mathcal{U} \to \mathcal{U}_b, \forall b.$$

Here, it is possible to *update each block separately* in order to reduce per-iteration computational costs and memory requirements, or to facilitate decentralization [Mao+20]. This corresponds to replacing the update in (8.1) by:

$$\forall b: \; u_{k+1,b} = u_{k,b} + \rho_{k,b}(T_b(u_k) - u_{k,b}), \tag{8.2}$$

where $\rho_{k,b}$ is a Boolean (random) variable that encodes if block $b$ is updated at iteration $k$.[1] Various strategies for selecting blocks are possible, such as cyclic updates or random sampling schemes. A generic convergence analysis of fixed-point iterations under both inexact and block updates has been proposed by Combettes and Pesquet [CP19], which we leverage in our analysis.

### 8.2.2 ADMM as a Fixed-Point Iteration

We now present how ADMM can be defined as a fixed-point iteration. ADMM minimizes the sum of two (possibly non-smooth) convex functions with linear constraints between the variables of these functions, which can be formulated as:

$$\begin{aligned} \underset{x,\, z}{\text{minimize}} \quad & f(x) + g(z) \\ \text{subject to} \quad & Ax + Bz = c \end{aligned} \tag{8.3}$$

ADMM is often presented as an approximate version of the augmented Lagrangian method, where the minimization of the sum in the primal is approximated by the alternating minimizations on $x$ and $z$. However, this analogy is not fruitful for theoretical analysis, as no proof of convergence only relies on bounding this approximation error to analyze ADMM [EY15]. A more useful characterization of ADMM is to see it as a splitting algorithm [EY15], i.e., an approach to find a fixed point of the composition of two (proximal) operators by performing operations that involve each operator separately. For completeness we recap the definition of proximal operator.

[1] Note that these block updates can be seen as projections of the global update and thus are also non-expansive.

**Definition 8.2** (Proximal operator)**.** *Let $f : \mathcal{U} \subset \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$ be a convex function that is lower semi-continuous and with its image not reduced to $\{+\infty\}$. The proximal operator $\operatorname{prox}_f$ is defined for $v \in \mathbb{R}^p$ by:*

$$\operatorname{prox}_f(v) = \arg\min_x \left( f(x) + \frac{1}{2}\|x - v\|^2 \right)$$

Specifically, ADMM can be defined through the Lions-Mercier operator [LM79]. Given two proximable functions $p_1$ and $p_2$ and parameter $\gamma > 0$, the Lions-Mercier operator is:

$$T_{\gamma p_1, \gamma p_2} = \lambda R_{\gamma p_1} R_{\gamma p_2} + (1 - \lambda) I, \tag{8.4}$$

where $R_{\gamma p_1} = 2 \operatorname{prox}_{\gamma p_1} - I$ and $R_{\gamma p_2} = 2 \operatorname{prox}_{\gamma p_2} - I$. This operator is $\lambda$-averaged, and it can be shown that if the set of the zeros of $\partial(f + g)$ is not empty, then the fixed points of $T_{\gamma p_1, \gamma p_2}$ are exactly these zeros [Boy+11].

The fixed-point iteration (8.1) with $T_{\gamma p_1, \gamma p_2}$ is known as the Douglas-Rachford algorithm, and ADMM is equivalent to this algorithm applied to a reformulation of (8.3) as $\min_u p_1(u) + p_2(u)$ with $p_1(u) = (-A \triangleright f)(-u - c)$ and $p_2(u) = (-B \triangleright g)(u)$, where we denote by $(M \triangleright f)(y) = \inf\{f(x) \mid Mx = y\}$ the infimal postcomposition [GB14]. For completeness, we show in Appendix E.2.1 how to recover the standard ADMM updates from this formulation.

Our analysis builds upon privacy amplification results. This includes *amplification by subsampling* [MTZ19]: if the above algorithm $A$ is executed on a random fraction $q$ of $\mathcal{D}$, then it satisfies $(\alpha, \mathcal{O}(\alpha \Delta^2 q^2 / 2\sigma^2))$-RDP. We also use *privacy amplification by iteration* [Fel+18; AT22]. This technique captures the fact that sequentially applying a non-expansive operator improves privacy guarantees for the initial point as the number of subsequent updates increase. Feldman et al. [Fel+18] and Altschuler and Talwar [AT22] applied this result to ensure differential privacy for SGD-type algorithms. *We use this result in tandem with the generic fixed-point iteration approach to develop and analyze the privacy of ADMM algorithms.*

## 8.3 A General Noisy Fixed-Point Iteration for Privacy Preserving Machine Learning

In this section, we formulate privacy preserving machine learning algorithms as instances of a general noisy fixed-point iteration. We show that we can recover popular private gradient descent methods (such as DP-SGD) from this formulation, and we provide a generic utility analysis.

### 8.3.1 Noisy Fixed-Point Iteration

**Algorithm 8.1:** Private fixed point iteration

**Input:** Non-expansive operator $R = (R_1, \dots, R_B)$ over $1 \leq B \leq p$ blocks, initial point $u_0 \in \mathcal{U}$, step sizes $(\lambda_k)_{k\in\mathbb{N}} \in (0, 1]$, active blocks $(\rho_k)_{k\in\mathbb{N}} \in \{0, 1\}^B$, errors $(e_k)_{k\in\mathbb{N}}$, privacy noise variance $\sigma^2 \geq 0$
**for** $k = 0, 1, \dots$ **do**
  **for** $b = 1, \dots, B$ **do**
    $u_{k+1,b} = u_{k,b} + \rho_{k,b}\lambda_k(R_b(u_k) + e_{k,b} + \eta_{k+1,b} - u_{k,b})$ with $\eta_{k+1,b} \sim \mathcal{N}(0, \sigma^2\mathbb{I}_p)$

Given a dataset $\mathcal{D} = (d_1, \dots, d_n)$, we aim to design differentially private algorithms to approximately solve the ERM problems of the form:

$$\underset{u \in \mathcal{U} \subseteq \mathbb{R}^p}{\text{minimize}} \quad \frac{1}{n}\sum_{i=1}^{n} f(u; d_i) + r(u), \tag{8.5}$$

where $f(\cdot; d_i)$ is a (typically smooth) loss function computed on data item $d_i$ and $r$ is a (typically non-smooth) regularizer. We denote $f(u; \mathcal{D}) \triangleq \frac{1}{n}\sum_{i=1}^{n} f(u; d_i)$.

To solve this minimization problem, we propose to consider the general noisy fixed-point iteration described in Algorithm 8.1. The core of each update applies a $\lambda_k$-averaged operator constructed from a non-expansive operator $R$, and a Gaussian noise term added to ensure differential privacy via the Gaussian mechanism (see Proposition 3.7). Algorithm 8.1 can use (possibly randomized) block-wise updates ($B > 1$) and accommodate additional errors in operator evaluation (in terms of $e_k$).

Despite the generality of this scheme, we show in Section 8.3.3 that we can provide provide a unified utility analysis under the only assumption that the operator $R$ is contractive.

### 8.3.2 Recovering Private Gradient-based Methods from the Noisy Fixed-Point Iteration

Differentially Private Stochastic Gradient Descent (DP-SGD) [BST14; Aba+16] is the most widely used private optimization algorithm (see Section 3.3.2). In Proposition 8.3, we show that we recover DP-SGD from our general noisy fixed-point iteration (Algorithm 8.1).

**Proposition 8.3** (DP-SGD as a noisy fixed-point iteration)**.** *Assume that $f(\cdot; d)$ is $\beta$-smooth for any $d$, and let $r(u) = 0$. Consider the non-expansive operator $R(u) \triangleq u - \frac{2}{\beta}\nabla f(u; \mathcal{D})$. Set $B = 1$, $\lambda_k = \lambda = \frac{\gamma\beta}{2}$ with $\gamma \in (0, \frac{2}{\beta})$, and $e_k = \frac{2}{\beta}(\nabla f(u_k) - \nabla f(u_k; d_{i_k}))$ with $i_k \in \{1, \dots, n\}$.[a] Then, Algorithm 8.1 recovers DP-SGD [BST14; Aba+16], i.e., the update at step $k+1$ is $u_{k+1} = u_k - \gamma(\nabla f(u_k; d_{i_k}) + \eta'_{k+1})$ with $\eta'_{k+1} \sim \mathcal{N}\left(0, \frac{\beta^2}{4}\sigma^2 I\right)$. The term $e_k$ corresponds to the error due*

*to evaluating the gradient on $d_{i_k}$ only, and satisfies $\mathbb{E}[\|e_k\|] \leq 4L/\beta$ when $f(\cdot; d)$ is $L$-Lipschitz for any $d$.*

[a] One can draw $i_k$ uniformly at random, or choose it so as to do deterministic passes over $\mathcal{D}$.

The privacy guarantees of DP-SGD can be derived: first, by observing that $R(u_k) + e_k = u_k - \nabla f_{i_k}(u_k; d_{i_k})$ is itself non-expansive, and then applying privacy amplification by iteration, as done in [Fel+18]. Alternatively, composition and privacy amplification by subsampling can be used [MTZ19].

Similarly, we also recover Differentially Private Coordinate Descent (DP-CD) [Man+22].

**Proposition 8.4** (DP-CD as a noisy fixed-point iteration)**.** *Consider the same setting as in Proposition 8.3, but with $B > 1$ blocks (coordinates), and $R_b(u) \triangleq u_b - \frac{2}{\beta}\nabla_b f(u; \mathcal{D})$, where $\nabla_b f$ is the $b$-th block of $\nabla f$, and $e_k = 0$. Then Algorithm 8.1 reduces to the Differentially Private Coordinate Descent (DP-CD) algorithm [Man+22].*

Utility guarantees for DP-SGD and DP-CD can be obtained as instantiations of the general convergence analysis of Algorithm 8.1, presented in Section 8.3.3.

### 8.3.3 Utility Analysis

In this section, we derive a utility result for our general noisy fixed-point iteration when the operator $R$ is *contractive* (see Definition 8.1). For gradient-based methods, this holds notably when $g$ is smooth and strongly convex. This is also the case for ADMM [see GB14; Ryu+20, and references therein for contraction constants under various sufficient conditions]. Our result, stated below, leverages a recent convergence result for inexact and block-wise fixed-point iterations [CP19]. The proof can be found in Appendix E.1.

**Theorem 8.5** (Utility guarantees for noisy fixed-point iterations)**.** *Assume that $R$ is $\tau$-contractive with fixed point $u^*$. Let $P[\rho_{k,b} = 1] = q$ for some $q \in (0, 1]$. Then there exists a learning rate $\lambda_k = \lambda \in (0, 1]$ such that the iterates of Algorithm 8.1 satisfy:*

$$\mathbb{E}\left(\|u_{k+1} - u^*\|^2 \mid \mathcal{F}_0\right) \leqslant \left(1 - \frac{q^2(1-\tau)}{8}\right)^k D + 8\left(\frac{\sqrt{p}\sigma + \zeta}{\sqrt{q}\,(1-\tau)} + \frac{p\sigma^2 + \zeta^2}{q^3(1-\tau)^3}\right)$$

*where $D \triangleq \|u_0 - u^*\|^2$, $p$ is the dimension of $u$, $\sigma^2 > 1 - \tau$ is the variance of the added Gaussian noise, and $\mathbb{E}[\|e_k\|^2] \leq \zeta^2$ for some $\zeta \geq 0$.*

Theorem 8.5 shows that our noisy fixed-point iteration enjoys a *linear convergence rate* up to an additive error term. The linear convergence rate depends on the contraction factor $\tau$ and the block activation probability $q$. The additive error term is ruled by the noise scale $\sigma\sqrt{p}+\zeta$, where $\sigma$ is due to the Gaussian noise added to ensure DP and $\zeta$ captures some possible additional error. Under a given privacy constraint, running more iterations requires to increase $\sigma$ (due to the composition rule of DP), yielding a classical privacy-utility trade-off ruled by the number of iterations. We investigate this in details for private ADMM algorithms in Section 8.4.

## 8.4 Private ADMM Algorithms

We now use our general noisy fixed-point iteration framework introduced in Section 8.3 to derive and analyze private ADMM algorithms for the centralized, federated and decentralized learning settings.

### 8.4.1 Private ADMM for Consensus

Given a dataset $\mathcal{D} = (d_1, \dots, d_n)$, we aim to solve an ERM problem of the form given in (8.5). This problem can be equivalently formulated as a consensus problem [Boy+11] that fits the general form (8.3) handled by ADMM:

$$\begin{aligned} \underset{x \in \mathbb{R}^{np},\, z \in \mathbb{R}^p}{\text{minimize}} \quad & \frac{1}{n}\sum_{i=1}^{n} f(x_i; d_i) + r(z) \\ \text{subject to} \quad & x - I_{n(p\times p)} z = 0, \end{aligned} \tag{8.6}$$

where $x = (x_1, \dots, x_n)^\top$ is composed of $n$ blocks (one for each data item) of size $p$ and $I_{n(p\times p)} \in \mathbb{R}^{np\times p}$ denotes $n$ stacked identity matrices of size $p \times p$. For convenience, we will sometimes denote $f_i(\cdot) \triangleq f(\cdot; d_i)$.

To privately solve problem (8.6), we apply our noisy fixed-point iteration (Algorithm 8.1) with the non-expansive operator $R_{\gamma p_1} R_{\gamma p_2}$ corresponding to ADMM (see Section 8.2.2). Introducing the auxiliary variable $u = (u_1, \dots, u_n) \in \mathbb{R}^{np}$ initialized to $u_0$ and exploiting the separable structure of the consensus problem (see Appendix E.2 for details), we obtain the following (block-wise) updates:

$$z_{k+1} = \operatorname{prox}_{\gamma r}\left(\tfrac{1}{n}\textstyle\sum_{i=1}^{n} u_{k,i}\right), \tag{8.7}$$

$$x_{k+1,i} = \operatorname{prox}_{\gamma f_i}(2z_{k+1} - u_{k,i}) \tag{8.8}$$

$$u_{k+1,i} = u_{k,i} + 2\lambda(x_{k+1,i} - z_{k+1} + \tfrac{1}{2}\eta_{k+1,i}). \tag{8.9}$$

**Algorithm 8.2:** Centralized private ADMM

**Input:** initial vector $u^0$, step size $\lambda \in (0, 1]$, privacy noise variance $\sigma^2 \geq 0$, $\gamma > 0$
**for** $k = 0$ *to* $K - 1$ **do**
  $\hat{z}_{k+1} = \frac{1}{n} \sum_{i=1}^{n} u_{k,i}$
  $z_{k+1} = \mathrm{prox}_{\gamma r}(\hat{z}_{k+1})$
  **for** $i = 1$ *to* $n$ **do**
    $x_{k+1,i} = \mathrm{prox}_{\gamma f_i}(2z_{k+1} - u_{k,i})$
    $u_{k+1,i} = u_{k,i} + 2\lambda(x_{k+1,i} - z_{k+1} + \frac{1}{2}\eta_{k+1,i})$ with $\eta_{k+1,i} \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_p)$
**return** $z_K$

From these updates and together with the possibility to randomly sample the blocks in our general scheme, we can naturally obtain different variants of ADMM for the centralized, federated and decentralized learning. In the remainder of this section, we present these variants, the corresponding trust models, and prove their privacy and utility guarantees.

**Remark 8.6** (General private ADMM)**.** *Our private ADMM algorithms for the consensus problem* (8.6) *are obtained as special cases of a private algorithm for the more general problem* (8.3)*. We present this algorithm in Appendix E.2.2. In Appendix E.3, we prove its privacy guarantees via a sensitivity analysis of the general update involving matrices $A$ and $B$, under the only hypothesis that $A$ is full rank. Then, we instantiate these general results to obtain privacy guarantees for private ADMM algorithms presented in this section.*

### 8.4.2 Centralized Private ADMM

In the centralized setting, a trusted curator holds the dataset $\mathcal{D}$ and seeks to release a model trained on it with record-level DP guarantees [CMS11]. Our private ADMM algorithm for this centralized setting closely follows the updates (8.7)-(8.9). The version shown in Algorithm 8.2 cycles over the $n$ blocks in a fixed order, but thanks to the flexibility of our scheme we can also randomize the choice of blocks at each iteration $k$, e.g., update a single random block or cycle over a random perturbation of the blocks. Note that at the end of the algorithm, we only release $z_K$, which is sufficient for all practical purposes. Returning $x_K$ would violate differential privacy as its last update interacts with the data through $\mathrm{prox}_{\gamma f_i}$ without subsequent random perturbation. The privacy guarantees of the algorithm are as follows.

**Theorem 8.7** (Privacy of centralized ADMM)**.** *Assume that the loss function $f(\cdot, d)$ is $L$-Lipschitz for any data record $d$ and consider record-level DP. Then Algorithm 8.2 satisfies $(\alpha, \frac{8\alpha K L^2 \gamma^2}{\sigma^2 n^2})$-RDP.*

*Sketch of proof.* We bound the sensitivity of the ADMM operator by relying on the structure of our updates, the strong convexity of proximal operators and known bounds on the sensitivity of the $\arg\min$ of strongly convex functions. The result then follows from composition. ■

Theorem 8.7 shows that the privacy loss of centralized ADMM has a similar form as that of state-of-the-art private gradient-based approaches like DP-SGD. The factor $K$ comes from the composition over the $K$ iterations, while the $L^2\gamma^2/n^2$ factor comes from the sensitivity of the ADMM operator. Crucially, the $1/n^2$ term allows for good utility when the number of data points is large enough. We also see that, similar to output perturbation [CMS11], the strong convexity parameter $1/\gamma$ of the proximal updates can be used to reduce the sensitivity.

By combining Theorem 8.7 and our generic utility analysis (Theorem 8.5 with $q = 1$), we obtain the following privacy-utility trade-off.

**Corollary 8.8** (Privacy-utility trade-off of centralized ADMM)**.** *Under the assumptions and notations of Theorem 8.5 and 8.7, setting $K$ appropriately, Algorithm 8.2 achieves*[a]

$$\mathbb{E}\left(\|u_K - u^*\|^2\right) = \widetilde{\mathcal{O}}\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon}n\,(1-\tau)} + \frac{p\alpha L^2\gamma^2}{\varepsilon n^2\,(1-\tau)^3}\right).$$

[a] $\widetilde{\mathcal{O}}$ ignores all the logarithmic terms.

### 8.4.3 Federated Private ADMM

We now switch to the Federated Learning (FL) setting [Kai+21] (see Section 2.3). We consider a set of $n$ users, with each user $i$ having a local dataset $d_i$ (which may consist of multiple data points). The function $f_i(\cdot) = f(\cdot; d_i)$ thus represents the local objective of user $i$ on its local dataset $d_i$. As before, we denote the joint dataset by $\mathcal{D} = (d_1, \ldots, d_n)$, but we now consider user-level DP.

Recall that in FL, the algorithm is orchestrated by a (potentially untrusted) central server and proceeds in rounds. Our federated private ADMM algorithm follows this procedure by essentially mimicking the updates of its centralized counterpart. Indeed, these updates can be executed in a federated fashion since (i) the blocks $x_i$ and $u_i$ associated to each user $i$ can be updated and perturbed locally and in parallel, and (ii) if each user $i$ shares $u_{k+1,i} - u_{k,i}$ with the server, then the latter can execute the rest of the updates to compute $z_{k+1}$. In particular, we do not need to send $x_i$ to the server during training (the consensus is achieved through $z$). On top of this vanilla version, we can natively accommodate *user sampling* (often called "client sampling" in the literature), which is a key property for cross-device FL as it allows to improve efficiency and to model partial user availability [Kai+21]. User sampling is readily obtained from our general scheme by choosing a subset of $m$ blocks (users) uniformly at random. Algorithm 8.3 gives the complete procedure.

**Algorithm 8.3:** Federated private ADMM

**Input:** initial point $z_0$, step size $\lambda \in (0, 1]$, privacy noise variance $\sigma^2 \geq 0$, parameter $\gamma > 0$, number of sampled users $1 \leq m \leq n$
**Server loop:**
**for** $k = 0$ *to* $K - 1$ **do**
  Subsample a set $S$ of $m$ users
  **for** $i \in S$ **do**
    $\Delta u_{k+1,i} = \textbf{LocalADMMstep}(z_k, i)$
  $\hat{z}_{k+1} = z_k + \frac{1}{n} \sum_{i \in S} \Delta u_{k+1,i}$
  $z_{k+1} = \mathrm{prox}_{\gamma r}(\hat{z}_{k+1})$
**return** $z_K$
**LocalADMMstep**$(z_k, i)$**:**
Sample $\eta_{k+1,i} \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_p)$
$x_{k+1,i} = \mathrm{prox}_{\gamma f_i}(2z_k - u_{k,i})$
$u_{k+1,i} = u_{k,i} + 2\lambda \left( x_{k+1,i} - z_k + \frac{1}{2}\eta_{k+1,i} \right)$
**return** $u_{k+1,i} - u_{k,i}$

The privacy guarantees of FL algorithms can be analyzed at two levels [NBD22]. The first level, corresponding to local DP [DJW13; Kas+08], is the privacy of each user with respect to the server (who observes the sequence of invidivual updates) or anyone eavesdropping on the communications. The second level, corresponding to central DP, is the privacy guarantee of users with respect to a third party observing only the final model. Our algorithm naturally provides these two levels of privacy, as shown in the following theorem.

**Theorem 8.9.** *Assume that the loss function $f(\cdot, d)$ is L-Lipschitz for any local dataset $d$ and consider user-level DP. Let $K_i$ be the number of participations of user $i$. Then, Algorithm 8.3 satisfies $(\alpha, \frac{8\alpha K_i L^2 \gamma^2}{\sigma^2})$-RDP for user $i$ in the local model. Furthermore, if $m < n/5$ and $\alpha \leq (M^2\sigma^2/2 - \log(5\sigma^2)) / (M + \log(m\alpha/n) + 1/(2\sigma^2))$ where $M = \log(1 + 1/(\frac{m}{n}(\alpha - 1)))$, then it also satisfies $(\alpha, \frac{16\alpha K L^2 \gamma^2}{\sigma^2 n^2})$-RDP in the central model.*

*Sketch of proof.* The local privacy guarantee follows from a sensitivity analysis, similarly to the centralized case. Then, we obtain the central guarantee by using amplification by subsampling and the aggregation of user contributions. ■

As expected, the local privacy guarantee does not amplify with the number of users $n$: since the server observes all individual updates, privacy only relies on the noise added locally by the user. In contrast, the central privacy guarantee benefits from both amplification by subsampling [MTZ19] thanks to user sampling (which gives a factor $m^2/n^2$) and by aggregation of the contributions of the $m$ sampled users (which gives a factor $1/m^2$). In the end, we thus recover the privacy guarantee of the centralized algorithm with the $1/n^2$ factor. We stress that the

restriction on $m/n$ and $\alpha$ in Theorem 8.9 is only to obtain the simple closed-form solution, as done in other works [see e.g. AT22]. In practice, privacy accounting is done numerically, see Appendix E.3.3 for details.

**Remark 8.10** (Secure aggregation)**.** *Our federated ADMM algorithm is compatible with the use of secure aggregation [Bon+17]. This allows the server to obtain $\sum_{i\in S}\Delta u_{k+1,i}$ without observing individual user contributions. In this case, the sensitivity is divided by $m$ and the privacy of users with respect to the server is thus amplified by a factor $1/m^2$. Therefore, for full participation $(m=n)$, we recover the privacy guarantee of the centralized case.*

We provide the privacy-utility trade-off by resorting to Theorem 8.5, where we fix $q = m/n = r$ with $r \in (0, 1/5]$.

**Corollary 8.11** (Privacy-utility trade-off of federated ADMM in the central model)**.** *Under the assumptions and notations of Theorem 8.5 and 8.9, setting $K$ appropriately, and also $m = rn$ for $r \in (0, 1/5)$, Algorithm 8.3 achieves*

$$\mathbb{E}\left\|u_K - u^*\right\|^2 = \widetilde{\mathcal{O}}\Big(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon r}n\left(1-\tau\right)} + \frac{p\alpha L^2\gamma^2}{\varepsilon r^2 n^2\left(1-\tau\right)^3}\Big).$$

### 8.4.4 Decentralized Private ADMM

Finally, we come back to the decentralized setting at the center of this thesis (see Section 2.4), focusing on the complete graph. Instantiating our general private ADMM algorithm with uniform subsampling of a single block at each iteration, we directly obtain a decentralized version of ADMM (Algorithm 8.4). The algorithm proceeds as follows. The model $z_0$ is initialized at some user $i$. Then, at each iteration $k$, the user with the model $z_k$ performs a local noisy update using its local dataset $d_i$, and then sends the resulting $z_{k+1}$ to a randomly chosen user. In other words, the model is updated by following a random walk. This random walk paradigm is quite popular in decentralized algorithms as seen in Chapter 7. In particular, it requires little computation and communication compared to other algorithms with more redundancy such as gossip algorithms see in Chapter 6.

It is easy to see that our decentralized algorithm enjoys the same local privacy guarantees as its federated counterpart (see Theorem 8.9). This provides a baseline protection against other users, and more generally against any adversary that would eavesdrop on all messages sent by the users. Yet, this guarantee can be quite pessimistic if the goal is to protect against other users in the system. Indeed, it is reasonable to assume that each user $i$ has only a limited view and only observes the messages it receives, without knowing the random path taken by the model between two visits to $i$. To capture this and improve privacy guarantees compared to the local model, we rely on the notion of *network DP*, as defined in Chapter 4.

**Algorithm 8.4:** Decentralized private ADMM

**Input:** initial points $u_0$ and $z_0$, step size $\lambda \in (0, 1]$, privacy noise variance $\sigma^2 \geq 0$, $\gamma > 0$
**for** $k = 0$ *to* $K - 1$ **do**
  Let $i$ be the currently selected user
  Sample $\eta_{k+1,i} \sim \mathcal{N}\left(0, \sigma^2 \mathbb{I}_p\right)$
  $x_{k+1,i} = \mathrm{prox}_{\gamma f_i}(2z_k - u_{k,i})$
  $u_{k+1,i} = u_{k,i} + 2\lambda \left(x_{k+1,i} - z_k + \frac{1}{2}\eta_{k+1,i}\right)$
  $\hat{z}_{k+1} = z_k + \frac{1}{n}(u_{k+1,i} - u_{k,i})$
  $z_{k+1} = \mathrm{prox}_{\gamma r}\left(\hat{z}_{k+1}\right)$
  Send $z_{k+1}$ to a random user

In our case, the view $\mathcal{O}_j$ of user $j$ is limited to $\mathcal{O}_j(A(\mathcal{D})) = (z_{k_l(j)})_{l=1}^{K_j}$ where $k_l(j)$ is the time of $l$-th contribution of user $j$ to the computation, and $K_j$ is the total number of times that $j$ contributed during the execution of algorithm. We can show the following network DP guarantees.

**Theorem 8.12.** *Assume that the loss function $f(\cdot, d)$ is $L$-Lipschitz for any local dataset $d$ and consider user-level DP. Let $\alpha > 1, \sigma > 2L\gamma\sqrt{\alpha(\alpha-1)}$ and $K_i$ the maximum number of contribution of a user. Then Algorithm 8.4 satisfies $(\alpha, \frac{8\alpha K_i L^2 \gamma^2 \log n}{\sigma^2 n})$-network RDP.*

*Sketch of proof.* Fixing a single participation of a given user (say $i$), we have the same local privacy loss as in the federated case. We then control how much this leakage decreases when the information reaches another user (say $j$). To do this, we first quantify the leakage when the $z$ variable is seen by user $j$ after $m$ steps by relying on privacy amplification by iteration. Then, thanks to the randomness of the path and the weak convexity of the Rényi divergence, we can average the different possible lengths $m$ of the path between users $i$ and $j$ in the complete graph. We conclude by composition over the number $K_i$ of participations of a user. ■

Remarkably, Theorem 8.12 shows that thanks to decentralization, we obtain a privacy amplification of $O(\log n/n^2)$ compared to the local DP guarantee. This amplification factor is of the same order as the one proved in Chapter 7 for a random walk version of DP-SGD, and matches the privacy guarantees of the centralized case up to a $\log n$ factor. To the best of our knowledge, this is the first result of this kind for ADMM, and the first application of privacy amplification by iteration to ADMM.

As before, we obtain the privacy-utility trade-off by resorting to Theorem 8.5, but this time with $q = 1/n$.

**Corollary 8.13** (Privacy-utility trade-off of decentralized ADMM)**.** *Under the assumptions and notations of Theorem 8.5 and 8.12, setting $K$ appropriately Algorithm 8.4 achieves*

$$\mathbb{E}\left(\|u_K - u^*\|^2\right) = \widetilde{\mathcal{O}}\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon n}\,(1-\tau)} + \frac{p\alpha L^2\gamma^2}{\varepsilon n\,(1-\tau)^3}\right).$$

**Remark 8.14** (Utility guarantees for centralized, federated, and decentralized settings.)**.** *From Corollary 8.8, we observe that the utility for the centralized setting is $\widetilde{\mathcal{O}}\left(\sqrt{\frac{p\alpha}{\varepsilon}}\frac{1}{n}\right)$ (in the regime $n \gg p$). On the other hand, the utility for the decentralized setting is $\widetilde{\mathcal{O}}\left(\sqrt{\frac{p\alpha}{\varepsilon n}}\right)$. This difference captures the shift in hardness from the centralized setting to the decentralized one. For the federated learning setting, the utility is $\widetilde{\mathcal{O}}\left(\sqrt{\frac{p\alpha}{\varepsilon mn}}\right)$, where $m$ is the number of sampled users at each step. Thus, if $m = n$ (all users contribute at each step), we recover the utility of the centralized setting. Instead, if $m = 1$, we are back to the utility of the decentralized setting. These observations demonstrate that our results on the privacy-utility trade-offs reasonably quantify the relative hardness of these three settings.*

## 8.5 Conclusion

In this chapter, we provide a unifying view of private optimization algorithms by framing them as noisy fixed-point iterations. The advantages of this novel perspective for privacy-preserving machine learning are at least two-fold. First, we give utility guarantees based only on very general assumptions on the underlying fixed-point operator, allowing us to cover many algorithms. Second, we show that we can derive new private algorithms by instantiating our general scheme with particular fixed-point operators. We illustrate this through the design of novel private ADMM algorithms for the centralized, federated and fully decentralized learning and the rather direct analysis of their privacy and utility guarantees. We note that an intrinsic limitation of our approach is that its generality may come at the cost of the tightness of utility guarantees, as we do not exploit the properties of specific algorithms.

We believe that our framework provides a general and principled approach to design and analyze novel private optimization algorithms by leveraging the rich literature on fixed-point iterations [CP21]. In future work, we would like to further broaden the applicability of our framework by proving (weaker) utility guarantees for $\lambda$-averaged operators that are non-expansive but not contractive. To achieve this, a possible direction is to extend the sublinear rates of [LFP15] to block-wise iterations.

# Chapter 9

# General Summary and Perspectives

## 9.1 Summary on our Contributions

In this thesis, we investigated how decentralization impacts privacy guarantees. In Chapter 4, we showed that the standard Decentralized Gradient Descent algorithm exposes nodes to reconstruction attacks, even when attackers and victims are separated by numerous nodes in the graph. This work demonstrated that decentralization alone does not ensure privacy unless combined with other defense mechanisms. In Chapter 5, we introduced two relaxations of differential privacy that build on the fact that participants only have local knowledge of the training and adapt the constraints to match the knowledge of a potential attacker. Subsequent chapters validated the soundness of this approach. The study of gossip algorithms in Chapter 6 illustrated that privacy guarantees are amplified by the accumulation of noise as distance grows in the graph, and that the computation of the privacy budget matches existing structures in the graph. It also showed that under this trust setting, the privacy guarantees nearly matches the ones obtained for the more stringent trust requirements of central differential privacy. In Chapter 7, we went further in this direction by showing that random walks, thanks to the randomness of the walks, achieve better privacy-utility trade-off than gossip algorithms when the spectral gap is large enough. In particular, we showed that we can explicitly relate the gossip matrix to the privacy guarantees of each pair of nodes, and drew connections with the existing graph-theoretic notion of communicability. Finally, in Chapter 8, we showed that it is possible to move beyond Stochastic Gradient Descent-based algorithms to design private optimization methods by leveraging the fixed-point iteration framework. Specifically, we presented a private ADMM algorithm that can benefit from privacy amplification mechanisms.

This thesis thus shows that interactions between graphs, algorithms, and privacy are complex and can exhibit a good synergy. By focusing on very general algorithmic formulations, we gain on generality but might be less tight than what has been achieved in follow-up works.

For instance, focusing on the ring topology, [Yak+22] were able to incorporate a detailed analysis of latency and stragglers. The idea that decentralization can, in some way, mimic a hidden state within an algorithm and thereby enhance privacy has inspired other works. For instance, [Bis+24] brings the idea of adding virtual nodes to the graph, [Gue+23] adds more randomness to walks by adding a probabilistic die out, [Lie+22] uses decentralization as a way to emulate a shuffling procedure and [All+24] improves over Muffliato's guarantees by relying on correlated noise. This thesis continue to open questions in several directions.

## 9.2 Perspectives

### 9.2.1 Follow-up Research Directions

**Vulnerability to Attacks in Decentralized Learning** Our attack uses three hypotheses: the graph is known, the private database is reduced to a single element, and the protocol is a synchronous D-GD based on gossip with rather stable gradients (i.e., near convergence or a small step size regime). It is likely that these three hypotheses can be successfully removed by designing more complex attacks. For the first, it would be interesting to combine this attack with an attack on graph topology [Che+21], and more broadly with existing literature on graph privacy, to determine whether we can infer enough information from the updates to reconstruct both the graph and the data. The second point is likely to be the easiest one, as one can directly take advantage of the progress in attacks on federated learning, such as [Boe+21; Kar+23], which could yield stronger results. Extending the attack to more complex algorithms such as random walks or randomized gossip is challenging: it is unclear if sufficient randomness emerges from these methods to effectively protect the private data of the nodes, or if they simply require more computational effort. Graphs are prone to combinatorial explosion, so providing an effective attack may seem implausible if the algorithm is sufficiently random. However, a training procedure with too much variance is also at odds with optimization goals, making it unclear whether a sweet spot exists where optimization remains effective but privacy attacks are mitigated. How to optimize the structure of the graph to minimize the number of reconstructible nodes, while still ensuring fast convergence, is an interesting open problem.

**Graph Optimization in Function of Pre-existing Trust Confidence** This graph optimization can be done not only from the "reconstructibility" point of view explained above, but also more generally as a graph optimization problem under various pairs of privacy budgets. As we are able to explicitly connect the gossip matrix $W$ to the privacy loss, optimizing $W$ is the natural next step. The ability to choose the graph can match practical scenarios. For instance, when considering a central server which communicates with all the nodes, it is possible to create a secure channel between any pair of nodes using a Diffie–Hellman key exchange. Similarly to

what is done in secure aggregation where a sparse but well-connected graph is used [Bel+20], it makes sense to design communication graphs depending on the affinity between nodes in the presence of an untrusted central server. In the longer term, this optimization process could be coupled with objectives beyond mere privacy, such as personalization based on similarities between nodes [VBT17] or mitigation of heterogeneity issues [LB+23]. Empowering the participants could stem from communication patterns designed to comply with local nodes' objectives rather than through a global optimization process.

**Extension to Other Algorithms** Our work proved the importance of the choice of algorithm for achieving privacy objectives: random walks do not yield the same profile of privacy budgets between nodes as synchronous gossip does. It is thus natural to explore new methods, as already investigated in the other works cited above. It seems important to introduce as much randomness as possible, while avoiding to pay a price in variance when doing so. An interesting direction would be to model more closely the randomness that appears in real-world decentralized systems, where asynchrony is the norm, dropout or missed packets are unavoidable, and communication needs to be compressed. These factors seem to present synergies with the privacy objective but could also introduce complexities not carefully modeled in our current representations of algorithms. Delving deeper in this direction could certainly bring decentralized differential privacy closer to real-world deployment. To create trust, it would also be important to make the guarantees given to users interpretable, which is often a hurdle in differential privacy.

### 9.2.2 Broader Outlook on Differential Privacy

We conclude this thesis with some thoughts on the future of differential privacy. It is quite common to cite the choice of the privacy budget in differential privacy as *the* open question to make DP practical. In my opinion, this question seems particularly useless and represents an unfair and irrelevant attack on DP: no one can hope that we will discover relevant rules that would be obvious to follow for every non-technical person without context[1]. The success of differential privacy is mainly explained by two properties: firstly, it provides many complex and beautiful mathematical questions, thus attracting researchers inclined towards theoretical and rigorous approaches of ethical AI. Secondly, it simplifies the abstract notion of privacy to a single scalar with the privacy budget. The privacy budget is easy to understand in simple instances: in the initial Randomized Response, it was understood by participants of sociology studies. As long as the output is discrete or low-dimensional, the probability histogram depending on the privacy budget is perfectly clear, and easier to understand than many machine learning metrics. The question becomes trickier in high-dimensional spaces where we cannot visualize

[1] If it does not ring a bell, you can go back to Section 3.1.

the output distribution. In such cases, DP takes a very protective stance by always bounding the maximum ratio between the most different distributions that could be generated, which means that it is always matches at least the protection we intuitively expect from a low-dimension intuition [BCH22]. Going beyond and allowing bigger budgets in some cases requires a better understanding of high-dimensional spaces and machine learning models, which are interesting questions not limited to differential privacy.

In the various applications which require higher privacy budgets, auditing and attacks provide good empirical insights into the concrete risks of reconstruction [Nas+21; CBP24; ZLS24; Nas+23]. Pursuing this direction improves our understanding and clarifies the desired order of magnitude for specific use cases. A privacy budget quantifies the complex notion of privacy, and thus suffers the same fate as other quantifiers: it will always be questioned and subject to interpretation, and cannot address all dimensions of privacy. A much more interesting question discussed and partially addressed in this thesis is the trust model and the granularity of the adjacency relationship.

When designing a privacy budget for ad placement, the most tangible risk for most people is overly effective personalization, leading them to spend more than necessary. In other words, no utility-privacy trade-off can be relevant, because the task itself is the privacy threat. The problem does not stem from the definition of DP; rather, it arises because the goal is inherently not private. Privacy washing in web tracking or video surveillance face the same limitations: if you destroy all utility when implementing privacy, maybe the task you are pursuing is an attack on privacy. In other contexts, however, this opposition is less direct: it should be possible to design machine learning models that detect tumors without memorizing the identities of patients who have cancer, or to predict global energy consumption needs at the scale of a neighborhood without learning the break patterns of every worker. In these cases, the exact value of the privacy budget is not extremely important. It should rather serve as an indicator to choose, among different methods, the one with the lowest risk, and the risk is likely to be manageable, as participants engage because they believe their involvement holds meaningful value. To be efficient, we should focus on designing tighter analyses, especially to track where privacy amplification can be found, by decentralizing some pre- or post-processing, by adding secure aggregation or shuffling. In other words, try to be sharp enough to distinguish between more and less private algorithms. Trust will not come from going from $\varepsilon = 19$ to $\varepsilon = 17$, but from the ability to join or stop participating in training, to have a say in the goals of the training and ensuring models with fair, robust and interpretable behaviors. Privacy is not an island of itself, it is a piece in the construction of trustworthy machine learning.

# List of Appendices

# Appendix A

# Proofs and Additional Results for Chapter 4

## A.1 Examples of Reconstructible Nodes

We illustrate the difficulty of knowing which nodes are reconstructible from explicit graph characteristics on some small examples reported in Figure A.1. The first graph $G1$ is the smallest one, but has the smallest proportion of reconstructible nodes $(2/4)$. Adding a single node as done in $G3$ makes all the nodes become reconstructible. However, adding a new node can also drastically reduce the proportion of reconstructible nodes, as shown in $G4$. Symmetry is not enough to prevent reconstruction, as highlighted in $G2$ where all nodes stay reconstructible.

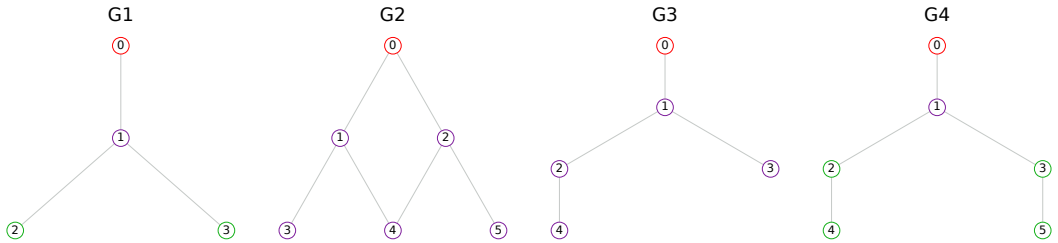



**Figure A.1** – Examples of similar graphs with different reconstructible sets. Attacker is node $0$ (red), reconstructible nodes are in purple, and non-reconstructible ones are in green.

## A.2 Construction of the Covariance Matrix for the Reconstruction Attack on Decentralized Gradient Descent

Algorithm A.1 shows how to build the covariance matrix $\Sigma^T$, necessary for the GLS method.

**Algorithm A.1:** Building the covariance matrix

**Input:** the graph $\mathcal{G}$, the set of attackers $\mathcal{A}$, the number of iterations $T$, $\sigma$ the noise amplitude
**Output:** The covariance matrix $\Sigma^T$
$\Sigma^T \leftarrow$ an empty matrix of size $m \times m$ where $m = T \cdot |\mathcal{N}(\mathcal{A})|$;
$i \leftarrow 0$;
**for** $t = 0$ **to** $T - 1$ **do**
  **foreach** $v \in \mathcal{N}(\mathcal{A})$ **do**
    $j \leftarrow t \cdot |\mathcal{N}(\mathcal{A})|$;
    **for** $t' = t$ **to** $T - 1$ **do**
      **foreach** $v' \in \mathcal{N}(\mathcal{A})$ **do**
        $C[i, j] \leftarrow \sigma^2 \left( \sum_{l=0}^{T} W_{\mathcal{T},\mathcal{T}}^{t+t'-2l} \right) [v, v']$;
        $C[j, i] \leftarrow C[i, j]$ ▷ * [r]Entry $(i, j)$ corresponds to the pair $((t, v), (t', v'))$
        $j \leftarrow j + 1$;
    $i \leftarrow i + 1$;
**return** $\Sigma^T$;

## A.3 Impact of Pairwise Relationship in Reconstruction

In this section, we report our findings on how the relationship between the attacker and its target affects the probability of reconstruction in gossip averaging. For a given attacker and target, the reconstruction can either be successful or failed, so it can be seen as a binary label. So how well can we predict this label based on the relation in the graph? This can be measured using Kendall rank correlation coefficient. We consider two relation metrics: the length of the shortest path between the attacker and the target, and their communicability [EH08], a measure used in graph mining. It is defined as a weighted average of the probability of going from one node to another in a given number of steps and measures how easy it is to communicate from one node to the other.

We use the same graphs as in the experiment on centrality, namely random Erdös Rényi graphs with $0$ as the attacker and the Facebook Ego graph $414$ where each node plays the role of the attacker in turn. We report the results in Table A.1. We see that in both cases, the shortest path length provides a good insight on the reconstruction probability, whereas the communicability only shows a small correlation.

| Relationship | Erdös-Rényi graph | Facebook Ego graph |
|---|---|---|
| Shortest Path Length | $-0.44 \pm 0.09$ | $-0.51 \pm 0.13$ |
| Communicability | $0.35 \pm 0.05$ | $0.08 \pm 0.35$ |

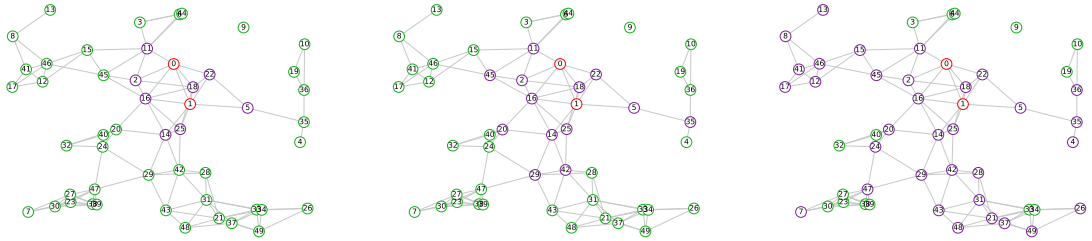
**Table A.1** – Kendall rank correlation coefficient between communicability, shortest paths, and the probability of reconstruction.

## A.4 Discussion of the Assumption of Public Knowledge of the Gossip Matrix

In our work, we assume that the attackers know the graph and the gossip matrix. While these quantities may not be fully known in some use-cases, we believe this is a justified assumption for the following reasons:

1. In many real-world scenarios, the graph topology is, or can be extracted from, public information. This is the case when nodes are hospitals in various universities (these collaborations are typically public knowledge) or financial institutions (e.g., SEC requires financial ties to be made public), in the context of computations over social network graphs such as Mastodon, and when learning within blockchain networks or distributed ledgers.

2. In general, it appears to be unsafe to assume that the graph and gossip matrix $W$ can be kept fully hidden from the attacker nodes. Since they are part of the learning process, attacker nodes must know at least their corresponding row. From this knowledge, with enough attacker nodes, they may be able to know/infer a large part of the graph. In particular, they can exploit the fact that $W$ must be doubly stochastic. Furthermore, keeping a graph private while publishing basic statistics (such as the spectral gap or the number of edges, triangles, or degree distribution) is also known to be hard. For instance, Chen et al. [Che+21, Section II.C therein] gives an example where a node can infer the edges of the graph from its own edges and the spectral gap; yet the spectral gap is often used to determine how many gossip steps are needed to reach a given precision. Therefore, given the challenge of precisely quantifying the risk of graph reconstruction, we find it reasonable to assume both the graph and matrix $W$ to be public information. This stance aligns with the established literature on differential privacy in decentralized learning, where it is commonly assumed that adversaries have access to the graph and matrix $W$ [see e.g. Cyf+22, and references therein].



(a) After 1 iteration (b) After 4 iterations (c) After 8 iterations

**Figure A.2** – Reconstruction after different number of steps of gossip averaging, on a random geometric graph of 50 nodes uniformly drawn from the unit square and a radius of 0.2. Attackers are in red, reconstructed nodes in purple, and non-reconstructed ones in green.

**Figure A.3** – Reconstruction attack on gossip averaging for several Facebook Ego graphs with different attackers chosen randomly. The node circled in red is the attacker, with reconstructed nodes shown in purple and non-reconstructed ones shown in yellow.

## A.5 Example of reconstruction on Random Geometric graphs

To illustrate the speed of reconstruction in gossip averaging, we report in Figure A.2 the reconstruction after 1 iterations, 4 iterations and 8 iterations (doing more iterations does not allow to reconstruct more nodes). Note that our attack support non-fully connected graph, although nodes that are part of different components are of course not reconstructed.

## A.6 Reconstruction Attacks on Facebook Ego Graphs

Figure A.3 shows the results of our reconstruction attack on gossip averaging for several Facebook Ego graphs. We report the same graphs twice but with different choices of attackers. This illustrates that the proportion of reconstructed nodes depends both on the graph structure

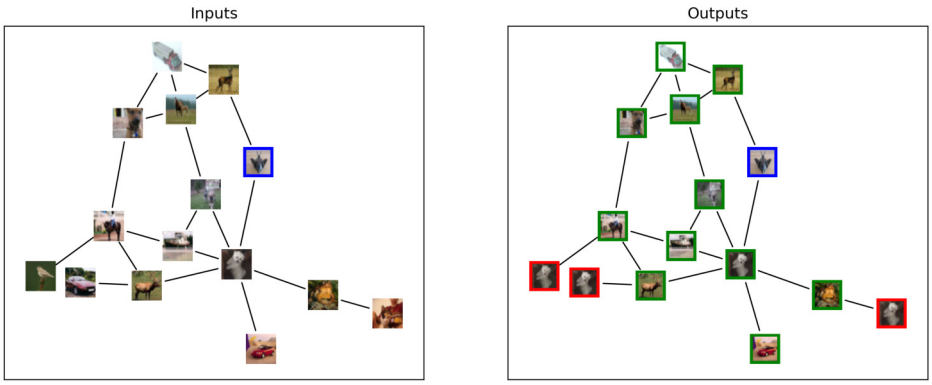



**Figure A.4** – Reconstruction attack on D-GD for the Florentine graph showing the true image inputs (left) and the reconstructions (right). The attacker is the node with the blue borders. Nodes with green borders are accurately reconstructed, the ones with red borders are not.

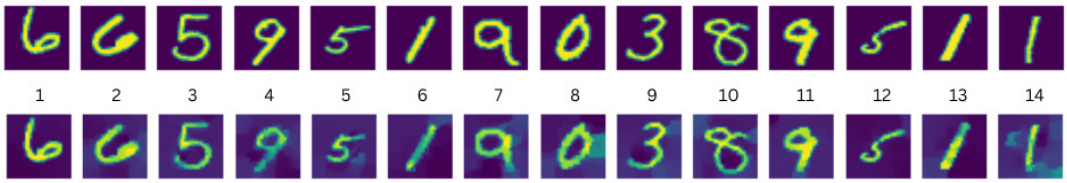



**Figure A.5** – Reconstruction results on D-GD for the Florentine graph. The first (resp. second) row shows the true (resp. reconstructed) inputs (Learning rate $10^{-5}$ and CNN model from Table A.2). The indices refer to the node labeling in Figure 4.5 where the attacker is node 0.

and the specific choice of the attacker. Note that in most cases, a vast majority of the nodes of the same community see their data leaked.

## A.7 Additional Experimental Results for Reconstruction Attacks on D-GD

In Figure A.4, we show the true inputs alongside the reconstructed images for the reconstruction attack.

In Figure A.5, we show an example of reconstruction on the Florentine graph with the MNIST dataset using a Convolutional Neural Network (details in Table A.2).

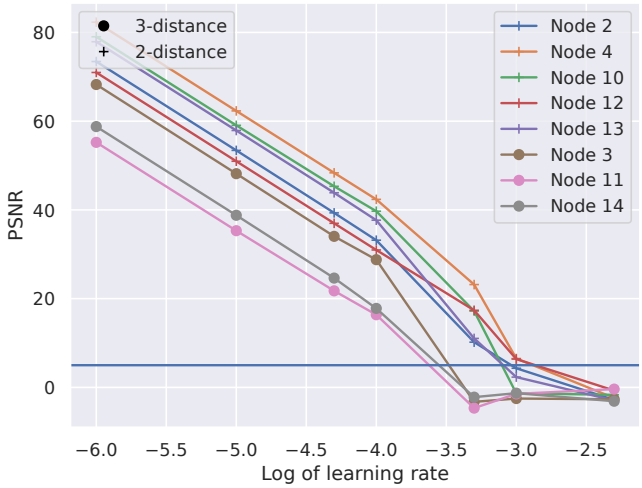



**Figure A.6** – Impact of the learning rate on the reconstruction attack on D-GD for the Florentine graph experiment shown in Figure 4.5. We plot the PSNR (averaged over 4 runs) between the reconstructed and original images for each of the target nodes and for different learning rates. We use different markers to classify the nodes that are at a distance of 2 from the attackers and the ones that are at a distance of 3 (The distance here refers to that of the shortest path between the attacker and the target). Points above the blue horizontal line are reconstructed with good visual quality.

## A.8 Details about the Convolutional Network

We report in Table A.2 the following Convolutional Neural Network for the experiments with MNIST.

| Layer Type | PyTorch Notation Size | Activation/Pooling |
|---|---|---|
| Convolution | (1, 32) | ReLU, Max Pooling |
| Convolution | (32, 64) | ReLU, Max Pooling |
| Fully Connected | (4096, 1024) | ReLU |
| Fully Connected | (1024, 10) | - |

**Table A.2** – Model Architecture Description

## A.9 Influence of the Learning Rate on the Attack on D-GD

In Figure A.6, we show the influence of the learning rate. When the learning rate becomes too large, gradients vary too much across iterations and it becomes impossible to make accurate reconstructions.

# Appendix B

# Proofs and Additional Results for Chapter 5

## B.1 Proof of Theorem 5.5 (Real Aggregation on a Ring)

*Proof.* We start by proving the utility claim. Algorithm 5.1 adds independent noise with standard deviation $\sigma_{loc}$ to the token every $n-1$ contributions. As there are $Kn$ steps, such noise is added $\lfloor Kn/(n-1) \rfloor$ times. By commutativity, the total noise has standard deviation $\sqrt{\lfloor Kn/(n-1) \rfloor}\sigma_{loc}$.

We now turn to the network differential privacy claim. Let us fix two distinct users $u$ and $v$ and consider what $v$ learns about the data of $u$. Recall that the structure of the ring is assumed to be public. The view $\mathcal{O}_v$ of $v$ (i.e., the information observed by $v$ during the execution of the protocol as defined in Eq. 5.1) thus corresponds to the $K$ values of the token that she receives. We denote these values by $\tau_1^v, \dots, \tau_K^v$, each of them corresponding to user contributions aggregated along with random noise. We define the view of $v$ accordingly as:

$$\mathcal{O}_v(\mathcal{A}(D)) = (\tau_i^v)_{i=1}^K. \tag{B.1}$$

Let us fix $i \in \{2, \dots, K\}$. By construction, $\tau_i^v - \tau_{i-1}^v$ is equal to the sum of updates between two visits of the token. In particular, we have the guarantee that at least one user different from $v$ has added noise in $\tau_i^v - \tau_{i-1}^v$ (as there are $n > n-1$ steps), and that $\tau_i^v - \tau_{i-1}^v$ does not contain more than one contribution made by $v$. It follows that the aggregation $\tau_{i+1}^v - \tau_i^v$ can be rewritten as $\text{Perturb}(x_u^i; \sigma_{loc}) + z$, where $z$ is independent from the contribution of $u$. By the $(\varepsilon, \delta)$-LDP property of $\text{Perturb}(\cdot; \sigma_{loc})$ and the post-processing property of differential privacy, we have for any $x, x'$:

$$\mathbb{P}(\tau_{i+1}^v - \tau_i^v = \tau | x_u^i = x) \leq e^\varepsilon \mathbb{P}(\tau_{i+1}^v - \tau_i^v = \tau | x_u^i = x') + \delta.$$

For the first token $\tau_1^v$, note it also contains noise with standard deviation $\sigma_{loc}$ added by the first user, so the same guarantee holds.

Finally, we apply the advanced composition theorem [DRV10] to get a differential privacy guarantee for the $K$ visits of the token, leading to the final privacy guarantee of $(\sqrt{2K\log(1/\delta')}\varepsilon + K\varepsilon(e^\varepsilon - 1), K\delta + \delta')$-network DP. ∎

## B.2 Proof of Theorem 5.7 (Histogram Computation on a Ring)

*Proof.* The proof is similar in spirit to the real summation case (see Appendix B.1), but leverages privacy amplification by subsampling to be able to quantify how much information is leaked by the value of the token (which is now a histogram).

We start by the utility claim (expected number of contributions). There are $Kn$ steps with at each step a probability $\gamma$ of adding a random response, plus the $\gamma n$ random responses at initialization, leading to a total of $\gamma n(K+1)$ random responses in expectation.

We now turn to the differential privacy guarantee. The view of a user $v$ is the content of the token at each visit of the token as defined in Eq. B.1, except that each $\tau_i^v \in \mathbb{N}^L$ is now a histogram over the domain $[L]$. More specifically, for $i \in \{2, \dots, K\}$, the difference $\tau_{i+1}^v - \tau_i^v$ between two consecutive tokens is now a discrete histogram of $n$ answers obtained by $RR_\gamma$ (each of them is random with probability $\gamma$). Similarly, in the first round, the token is initialized with $\gamma n$ random elements. Therefore, we can apply results from amplification by shuffling, because a discrete histogram carries the same more information as a shuffle of the individual values. In particular, we can use Corollary 9 from [Erl+19] that we recall below.

**Theorem B.1** (Erlingsson). *Let $n \geq 100, 0 < \varepsilon_0 < \frac{1}{2}$ and $\delta < \frac{1}{100}$. For a local randomizer ensuring $\varepsilon_0$-LDP, the shuffling mechanism is $(\varepsilon, \delta)$-differentially private with*

$$\varepsilon = 12\varepsilon_0\sqrt{\frac{\log(1/\delta)}{n}}.$$

We can apply this result to the information revealed by the value of the token between two visits to user $v$. The required LDP guarantee is ensured by the use of the randomized response mechanism, where we set $\gamma$ so that $RR_\gamma$ satisfies $12\varepsilon\sqrt{\frac{\log(1/\delta)}{n}}$-LDP, leading to an $(\varepsilon, \delta)$-DP guarantee after shuffling. We conclude by the application of advanced composition [DRV10]. ■

## B.3 Relation between Network DP on a Ring and Pan-Privacy

In this section, we highlight an interesting connection between the specific case of network DP on a ring topology and the pan-privacy model [Dwo+10]. In the pan-privacy model, raw data is processed in an online fashion by a central party. This central party is trusted to process raw data but not to store it in perpetuity, and its storage may be subject to breaches (i.e., its internal state may become visible to an adversary). It can thus be seen as an intermediate trust model between the central and local models. Connections between pan-privacy and the shuffle model have been recently studied [Bal+20a], allowing in some cases to adapt algorithms from one setting to the other. Other recent work has studied the relation between pan-privacy with several breaches and the local model [AJM19].

To formally define pan-privacy, we first need to define what we mean by online algorithms. An online algorithm receives a stream of raw data and sequentially updates an internal state

with one data point before deleting it. At the end of the stream, the algorithm publishes a final output based on its last internal state.

**Definition B.2** (Online algorithm)**.** *An online algorithm $\mathcal{A}$ is defined by a sequence of internal algorithms $\mathcal{A}_1, \dots$ and an output algorithm $\mathcal{A}_O$. Given an input stream $\vec{x}$, the first internal algorithm $\mathcal{A}_1 : \mathcal{X} \to \mathcal{I}$ maps $x_1$ to a state $s_1$, and for $i \geq 2$, $\mathcal{A}_i : \mathcal{X} \times \mathcal{I} \to \mathcal{I}$ maps $x_i$ and the previous state $s_{i-1}$ to a new state $s_i$. At the end of the stream, $\mathcal{A}$ publishes a final output by executing $\mathcal{A}_O : \mathcal{I} \to O$ on its final internal state. We denote by $\mathcal{A}_{\mathcal{I}}(\vec{x})$ the internal state of $\mathcal{A}$ after processing stream $\vec{x}$.*

In pan-privacy, the algorithm is trusted to process a raw data stream, but should protect its internal states against potential breaches. The moment of the update where the state is modified by a raw data point is supposed to be atomic. Hence, the observable impact of a data point is restricted to the internal state and the final output. Below, we state the standard definition of pan-privacy with a single breach, i.e., the adversary may observe a single internal state in addition to the final output. Two streams $\vec{x}, \vec{x}'$ are said to be neighboring if they differ in at most one element.

**Definition B.3** (Pan-privacy)**.** *An online algorithm $\mathcal{A}$ is $(\varepsilon, \delta)$-pan private if for every pair of neighboring streams $\vec{x} \sim \vec{x}'$, for every time $t$, and for every subset $T \subseteq \mathcal{I} \times \mathcal{O}$, we have:*

$$\mathbb{P}\left(\left(\mathcal{A}_{\mathcal{I}}\left(\vec{x}_{\leq t}\right), \mathcal{A}_O\left(\mathcal{A}_{\mathcal{I}}(\vec{x})\right)\right) \in T\right) \leq e^{\varepsilon} \cdot \mathbb{P}\left(\left(\mathcal{A}_{\mathcal{I}}\left(\vec{x}'_{\leq t}\right), \mathcal{A}_O\left(\mathcal{A}_{\mathcal{I}}\left(\vec{x}'\right)\right)\right) \in T\right) + \delta,$$

*where $\vec{x}_{\leq t}$ denotes the first $t$ elements of stream $\vec{x}$.*

We can now make a connection between the above pan-privacy definition and our simple protocols for network DP on a ring topology introduced in Section 5.4, in the case where each user contributes only once ($K = 1$ in our notations). In our network DP setting, the internal state corresponds to the value of the token and the final output is empty (or is equal to the final state of the token, if one performs an additional cycle over the ring during which the token is left unchanged to broadcast it to all users). A breach at time $t$ (i.e., observation of internal state $s_t$) corresponds to the observation of the token by the $t$-th user. Note also that our neighboring relation on the users' datasets is equivalent to that on data streams for the case of $K = 1$. Therefore, we can simulate a pan-private algorithm as a network DP algorithm on a ring.

We note that the lack of privacy gains for network DP compared to local DP when considering a ring topology with collusions (see discussion in Section 5.4.3) is in line with the reduction of [AJM19], which shows that in pure pan-privacy, protection against multiple breaches is equivalent to sequentially interactive local privacy.

While network DP reduces to pan-privacy when the topology is the ring and one considers simple protocols with a single token and a single contribution per user, we emphasize that our model is more general and potentially allows superior privacy-utility trade-offs for more complex protocols and/or topologies. This is illustrated by our results on the complete graph, where breaches cannot follow an arbitrary pattern. Indeed, as a breach corresponds to sending the token to a colluding user, this risk is mitigated by the properties of the random walk: as long as the token is held by non-colluding users, the walk stays unbiased and thus does not return too quickly to colluding users. This additional structure on the potential breaches give us the room for stronger guarantees.

# Appendix C

# Proofs and Additional Results for Chapter 6

## C.1 Preliminary Lemmas and Notations

We conduct our analysis with the Gaussian mechanism that we recall here for readability.

**Lemma C.1** (Gaussian mechanism). *For $\alpha > 1$, noise variance $\sigma^2$, sensitivity $\Delta > 0$ and $x, y \in \mathbb{R}$ such that $|x - y| \leq \Delta$, we have:*

$$D_\alpha(\mathcal{N}(x, \sigma^2) \,||\, \mathcal{N}(y, \sigma^2)) \leq \frac{\alpha \Delta^2}{2\sigma^2} \,.$$

For the privacy analysis when the graph is private and randomly sampled (Appendix C.5), we use the following result on the weak convexity of the Rényi divergence [Fel+18].

**Lemma C.2** (Quasi-convexity of Rényi divergence [Fel+18]). *Let $(\mu_i)_{i\in\mathcal{I}}$ and $(\nu_i)_{i\in\mathcal{I}}$ be probability distributions over shared space, such that for all $i \in \mathcal{I}$, we have $D_\alpha(\mu_i||\nu_i) \leq c/(\alpha - 1)$ for some $c \in (0, 1]$. Let $\rho$ be a distribution over $\mathcal{I}$ and $\mu_\rho$ and $\nu_\rho$ be obtained by sampling $i$ from $\rho$, and outputting a sample from $\mu_i$ and $\nu_i$. Then, we have:*

$$D_\alpha(\mu_\rho||\nu_\rho) \leq (1 + c)\mathbb{E}\left[D_\alpha(\mu_i||\nu_i) \mid i \sim \rho\right] \,.$$

In the following, we will use the notation $u \sim v$ to denote that two nodes $u$ and $v$ are neighbors.

## C.2 General Privacy Analysis (Theorem 6.1)

*Proof.* We need to bound the privacy loss that occurs from the following view:

$$\mathcal{O}_v\left(\mathcal{A}^T(\mathcal{D})\right) = \{(W_{0:t}(x + \eta))_w \mid \quad \{v, w\} \in \mathcal{E}_t \,, \quad 0 \leq t \leq T - 1\} \cup \{x_v\} \,.$$

We have:

$$D_\alpha(\mathcal{O}_v(\mathcal{A}^T(\mathcal{D})) \,||\, \mathcal{O}_v(\mathcal{A}^T(\mathcal{D}'))) \leq \sum_{t=0}^{T-1} \sum_{w\in\mathcal{N}_t(v)} D_\alpha\big(\,(W_{0:t}(x + \eta))_w \,||\, (W_{0:t}(x' + \eta))_w\,\big) \,.$$

We have $(W_{0:t}(x' + \eta))_w - (W_{0:t}(x + \eta))_w \sim \mathcal{N}((W_{0:t}(x' + \eta))_w - (W_{0:t}(x + \eta))_w, \sigma^2\|(W_{0:t})_w\|^2)$ with a sensitivity verifying $|(W_{0:t}(x' + \eta))_w - (W_{0:t}(x + \eta))_w|^2 \leq \Delta^2 (W_{0:t})^2_{u,w}$ with $\Delta^2$ the sensitivity and $\mathcal{D} \sim_u \mathcal{D}'$. Thus, using Lemma C.1, we have:

$$D_\alpha\big(\,(W_{0:t}(x + \eta))_w \,||\, (W_{0:t}(x' + \eta))_w\,\big) \leq \frac{\alpha\Delta^2}{2\sigma^2} \frac{(W_{0:t})^2_{u,w}}{\|(W_{0:t})_w\|^2} \,,$$

leading to the desired $f(u,v)$. The mean privacy loss is then obtained by summing the above inequality for $u \neq v$ and $t < T$:

$$\begin{aligned}
\bar{\varepsilon}_v = \frac{1}{n}\sum_{u\neq v} f(u,v) &\leq \frac{1}{n}\sum_{u\in\mathcal{V}} \frac{\alpha\Delta^2}{2\sigma^2} \sum_{w\in\mathcal{V}} \sum_{t\in\mathcal{P}^T_{\{v,w\}}} \frac{(W_{0:t})^2_{u,w}}{\|(W_{0:t})_w\|^2} \\
&= \frac{1}{n}\frac{\alpha\Delta^2}{2\sigma^2} \sum_{t\in\mathcal{P}^T_{\{v,w\}}} \sum_{w\in\mathcal{V}} \sum_{u\in\mathcal{V}} \frac{(W_{0:t})^2_{u,w}}{\|(W_{0:t})_w\|^2} \\
&= \frac{1}{n}\frac{\alpha\Delta^2}{2\sigma^2} \sum_{t\in\mathcal{P}^T_{\{v,w\}}} \sum_{w\in\mathcal{V}} 1 \\
&= \frac{\alpha\Delta^2 T_v}{2n\sigma^2}\,,
\end{aligned}$$

where $T_v = \sum_{w\in\mathcal{V}} |\mathcal{P}^T_{\{v,w\}}|$ is exactly the number of communications node $v$ is involved in, up to time $T$. ■

## C.3 Synchronous *Muffliato*

### C.3.1 Utility Analysis (Theorem 6.2)

While the main text presents the result for the 1-dimensional case for simplicity, we prove here the convergence for the general case where each node holds a vector of dimension $D$. Theorem 6.2 is then a direct consequence of Equation (C.1) for $D = 1$.

**Theorem C.3** (Utility analysis). *For any $T \geq 0$, the iterates $(x^T)_{T\geq 0}$ of* Muffliato *(Algorithm 6.1) verify, for $\lambda_W$ the spectral gap and $\bar{x} = \frac{1}{n}\sum_{v\in\mathcal{V}} x_v$:*

$$\frac{1}{2n}\sum_{v\in\mathcal{V}} \mathbb{E}\left[\left\|x_v^T - \overline{x}\right\|^2\right] \leq \left(\frac{1}{n}\sum_{v\in\mathcal{V}} \|x_v - \overline{x}\|^2 + D\sigma^2\right) e^{-T\sqrt{\lambda_W}} + \frac{D\sigma^2}{n}\,. \tag{C.1}$$

*Proof.* For $t \geq 0$ and $y \in \mathbb{R}^{\mathcal{V}\times D}$, using results from Berthier, Bach, and Gaillard [BBG20], we have, for a vector $y \in \mathbb{R}^{\mathcal{V}\times D}$ such that $\sum_{v\in\mathcal{V}} y_v = 0$, $\|P_t(W)y\|^2 \leq 2(1-\sqrt{\lambda_W})^2\|y\|^2$. In particular:

$$\left\|P_t(W)(y - \overline{y}\mathbb{1}^\top)\right\| \leq 2(1-\sqrt{\lambda_W})^t \left\|y - \overline{y}\mathbb{1}^\top\right\|^2,$$

where $\mathbb{1}$ with the vector with all entries equal to 1. Since

$$x^t = P_t(W)\left(x + \mathcal{N}\left(0, \sigma^2 I_{\mathcal{V}\times D}\right)\right)\,, \quad t \geq 0\,,$$

we obtain that, for $\eta \sim \mathcal{N}\left(0, \sigma^2 I_{\mathcal{V}\times D}\right)$ and $\overline{\eta} = \frac{1}{n}\sum_{v\in\mathcal{V}} \eta_v \mathbb{1}^\top \in \mathbb{R}^{\mathcal{V}\times D}$, using bias-variance decomposition twice:

$$\begin{aligned}
\frac{1}{2}\mathbb{E}\left[\left\|x^t - \overline{x}\right\|^2\right] &= \frac{1}{2}\mathbb{E}\left[\left\|P_t(W)(x^{(0)} - \overline{x})\right\|^2\right] \\
&= \frac{1}{2}\|P_t(W)(x + \eta - \overline{x} - \overline{\eta})\|^2 + \frac{1}{2}\mathbb{E}\left[\|P_t(W)\overline{\eta}\|^2\right] \\
&\leq (1 - \sqrt{\lambda_W})^t \mathbb{E}\left[\|x + \eta - \overline{x} - \overline{\eta}\|^2\right] + \frac{D\sigma^2}{2n} \\
&\leq (1 - \sqrt{\lambda_W})^t \left(\mathbb{E}\left[\|x - \overline{x}\|^2\right] + nD\sigma^2\right) + \frac{D\sigma^2}{2n}\,. \qquad \blacksquare
\end{aligned}$$

The precision $\frac{3D\sigma^2}{n}$ is thus reached for

$$T^{\mathrm{stop}}\left(W, (x_v)_{v\in\mathcal{V}}, D\sigma^2\right) \leq \sqrt{\lambda_W}^{-1} \log\left(\frac{n}{D\sigma^2}\max\left(D\sigma^2, \frac{1}{n}\sum_{v\in\mathcal{V}}\|x_v - \overline{x}\|^2\right)\right)\,.$$

### C.3.2 Privacy Analysis (Corollary 6.3)

*Proof of Corollary 6.3.* For a fixed gossip matrix $W$, we have $W_{0:t} = W^t$, so that Theorem 6.1 reads:

$$f(u,v) = \frac{\alpha\Delta^2}{2\sigma^2}\sum_{t<T}\sum_{w:\{v,w\}\in\mathcal{E}} \frac{(W^t)^2_{u,w}}{\|(W^t)_w\|^2}\,.$$

Since $W$ is bi-stochastic, $\frac{(W^t)^2_{u,w}}{\|(W^t)_w\|^2} \leq n \times (W^t)^2_{u,w} = n\mathbb{P}\left(X^t = u | X^0 = w\right)^2$, using Cauchy-Schwarz inequality.

Summing over the neighbors $w \sim v$, we obtain, for $t < T$:

$$\begin{aligned}
\sum_{w\sim v}\frac{\alpha}{2\sigma^2}\sum_{w:\{v,w\}\in\mathcal{E}}\frac{(W^t)^2_{u,w}}{\|(W^t)_w\|^2} &\leq \frac{\alpha n}{2\sigma^2}\sum_{w\sim v}\mathbb{P}\left(X^t = u | X^0 = w\right)^2 \\
&\leq \frac{\alpha n}{2\sigma^2}\left(\sum_{w\sim v}\mathbb{P}\left(X^t = u | X^0 = w\right)\right)^2 \\
&\leq \frac{\alpha n}{2\sigma^2}\frac{1}{\min_{w\sim v} W^2_{v,w}}\left(\sum_{w\sim v} W_{v,w}\mathbb{P}\left(X^t = u | X^0 = w\right)\right)^2 \\
&\leq \frac{\alpha n}{2\sigma^2}\frac{1}{\min_{w\sim v} W^2_{v,w}}\mathbb{P}\left(X^{t+1} = u | X^0 = v\right)^2\,,
\end{aligned}$$

where the last line is obtained by observing that:

$$\sum_{w\sim v} W_{v,w}\mathbb{P}\left(X^t = u | X^0 = w\right) = \mathbb{P}\left(X^{t+1} = u | X^0 = v\right)\,,$$

by conditioning on the first step of the random walk. This leads to Corollary 6.3. ■

### C.3.3 First Line of Table 6.1

The results in the first line of Table 6.1 are obtained by observing that $v$ is involved in $d_v T$ communications up to time $T$, leading to $\bar{\varepsilon}_v = \frac{\alpha\Delta^2 d_v T}{2\sigma^2 n^2}$. Using Theorem 6.2, we have a utility of $3\sigma^2/n$ for $T^{\mathrm{stop}} = \lambda_W^{-1/2} \log\left(\frac{n}{\sigma^2}\max\left(\sigma^2, \frac{1}{n}\sum_{v\in\mathcal{V}}\left\|x_v^0 - \overline{x}\right\|^2\right)\right)$ steps. Thus, imposing $\bar{\varepsilon}_v \leq \varepsilon$ for a fixed $\varepsilon > 0$ gives us $\sigma^2 = \frac{\alpha\Delta^2 d T^{\mathrm{stop}}}{2\sigma^2 n^2}$, leading to a utility of

$$\tilde{\mathcal{O}}\Big(\frac{\alpha\Delta^2 d}{2\sigma^2\sqrt{\lambda_W}}\Big)\,.$$

We then instantiate this formula on graphs with known spectral gaps, as described for instance in Mohar et al. [Moh+91].

## C.4 Randomized *Muffliato*

### C.4.1 Utility Analysis (Theorem 6.4)

As in the synchronous case, we prove a more general convergence result that holds for $D$-dimensional inputs. Then, Theorem 6.4 follows directly from (C.2).

**Theorem C.4** (Utility analysis)**.** *For any $T \geq 0$, the iterates $(x^T)_{T\geq 0}$ of randomized* Muffliato *(Algorithm 6.2) verify:*

$$\frac{1}{2n}\sum_{v\in\mathcal{V}}\mathbb{E}\left[\left\|x_v^T - \overline{x}\right\|^2\right] \leq \left(\frac{1}{n}\sum_{v\in\mathcal{V}}\left\|x_v^0 - \overline{x}\right\|^2 + D\sigma^2\right)e^{-T\lambda_2(p)} + \frac{D\sigma^2}{n}\,. \tag{C.2}$$

*Proof of Theorem 6.4.* For $t \geq 0$ and $y \in \mathbb{R}^{\mathcal{V}\times D}$, using results from Boyd et al. [Boy+06], we have:

$$\mathbb{E}\left[\left\|W(t)(y - \overline{y}\mathbb{1}^\top)\right\|^2\right] \leq (1-\lambda(p))^t\left\|y - \overline{y}\mathbb{1}^\top\right\|^2,$$

where $\mathbb{1}$ with the vector with all entries equal to 1 and $\overline{y} = \frac{1}{n}\sum_{v\in\mathcal{V}} y_v$. The rest of the proof follows as in the proof of Theorem 6.2 with two bias-variance decompositions. ■

The precision $\frac{2D\sigma^2}{n}$ is thus reached for

$$T^{\mathrm{stop}}\left(W, (x_v)_{v\in\mathcal{V}}, \sigma^2\right) \leq \lambda(p)^{-1}\log\left(\frac{n}{D\sigma^2}\max\left(D\sigma^2, \frac{1}{n}\sum_{v\in\mathcal{V}}\|x_v - \overline{x}\|^2\right)\right)\,. \tag{C.3}$$

### C.4.2 Privacy Analysis

In terms of privacy, randomized *Muffliato* satisfies the following guarantees, obtained by applying Theorem 6.1.

**Corollary C.5.** *After $T$ iterations of randomized* Muffliato*, and conditionally on the edges sampled, node $u \in \mathcal{V}$ is $(\alpha, \varepsilon^T_{u\to v}(\alpha))$-PNDP with respect to $v$, with:*

$$\varepsilon^T_{u\to v}(\alpha) \leq \frac{\alpha\Delta^2}{2\sigma^2} \sum_{w\sim v} \sum_{t\in\mathcal{P}^T_{\{v,w\}}} \frac{(W_{0:t})^2_{uw}}{\|(W_{0:t})_w\|^2} \,.$$

*Taking the mean over $u \neq v$ yields:*

$$\bar{\varepsilon}_v = \frac{1}{n} \sum_{u\in\mathcal{V}\setminus\{v\}} \varepsilon^T_{u\to v}(\alpha) \leq \frac{\alpha\Delta^2}{2n\sigma^2} T_v \,,$$

*where $T_v = \sum_{t<T} \sum_{w\sim v} \mathbb{1}_{\{\{v,w\}=\{v_t,w_t\}\}}$ the number of communications node $v$ is involved in up to time $T$.*

Note that $T_v$ is a Binomial random variable of parameters $(T, \pi_v)$ where $\pi_v = \sum_{w\sim v} p_{\{v,w\}}$.

We now explain how we obtain the second line of Table 6.1.

Note that a choice $p_{\{v,w\}} = 2W_{v,w}/n$ for some given gossip matrix $W$ yields probability activations. For the sake of comparison with *Muffliato* with a fixed matrix, we place ourselves in this case. This leads to $\pi_v = 2/n$, so that

$$\mathbb{E}\left[\bar{\varepsilon}_v\right] = \frac{\alpha\Delta^2 T}{2n^2\sigma^2} \,,$$

and for any $C > 0$,

$$\mathbb{P}\left(T_v - \mathbb{E}T_v \geq C\right) \leq \exp\left(-\frac{C^2}{T}\right) \,,$$

using Hoeffding's inequality. We take as time-horizon $T = T^{\text{stop}} \geq 1/\lambda(p)$ defined in Theorem 6.4, leading to

$$\mathbb{P}\left(T_v - \mathbb{E}T_v \geq C\right) \leq \exp\left(-\frac{C^2\lambda_W}{n}\right) \,, \quad \mathbb{E}\left[\bar{\varepsilon}_v\right] = \tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{2n\sigma^2\lambda_W}\right) \,,$$

since $\lambda(p) = \frac{2\lambda_W}{n}$ in our case.

The same methodology as in the synchronous case (imposing $\bar{\varepsilon}_v \leq \varepsilon$ for the time horizon $T^{\text{stop}}$, deriving $\sigma^2$ from this and thus the resulting utility) leads to the second line of Table 6.1.

## C.5 *Muffliato* on Private Random Graphs (Theorem 6.6)

We fix all nodes, and in particular $u$ the attacked node, and $v$ the observer. We assume that $G$ is drawn randomly. Edges $\{v, w\}$ are drawn independently from one another. The result we prove below is just slightly more general than Theorem 6.6 that is recovered for $\mathbb{P}(\{u, v\} \in \mathcal{E}) = q$ (Erdös-Rényi random graph).

We make the following assumption: node $v$ is only aware of its direct neighbors in the topology of graph $G$, and conditionally on $\{v\} \cup \mathcal{N}(v)$, the law of the graph is invariant under any permutation over the set $\mathcal{V} \setminus (\{v\} \cup \mathcal{N}(v))$.

**Theorem C.6** (*Muffliato* with a random graph)**.** *Let us fix nodes $u$ and $v$. Let $\alpha > 1$, $T \geq 0$, $\sigma^2 \geq \frac{\Delta^2 \alpha(\alpha-1)}{2}$ and let the above assumptions on $G$ be satisfied. After running Algorithm 6.1 with these parameters, node $u$ is $(\alpha, \varepsilon^T_{u\to v}(\alpha))$-PNDP with respect to $v$, with:*

$$\varepsilon^T_{u\to v}(\alpha) \leq \begin{cases} \dfrac{\alpha\Delta^2}{2\sigma^2} & \text{with probability } \mathbb{P}(\{u, v\} \in \mathcal{E}) \ , \\ \dfrac{\alpha\Delta^2}{\sigma^2}\dfrac{Td_v}{n - d_v} & \text{with probability } 1 - \mathbb{P}(\{u, v\} \in \mathcal{E}) \ . \end{cases}$$

*Proof.* If $\{u, v\} \in \mathcal{E}$, we cannot do better than $\varepsilon^T_{u\to v} \leq \frac{\alpha\Delta^2}{2\sigma^2}$: $v$ only sees $x_u^{(0)} + \mathcal{N}(0, \sigma^2)$ and then next messages can be seen as post-processing of this initial message and thus do not induce further loss. This happens with probability $\mathbb{P}(\{u, v\} \in \mathcal{E})$.

Now, we reason conditionally on $\{v\} \cup \mathcal{N}(v)$ and $u \notin \mathcal{N}(v)$. Using the averaged formula (6.3), we have:

$$\frac{1}{n}\left(\sum_{w\in\mathcal{N}(v)} \frac{\alpha\Delta^2}{2\sigma^2} + \sum_{w\in\mathcal{V}\setminus(\mathcal{N}(v)\cup\{v\})} \varepsilon^T_{w\to v}(\alpha)\right) \leq \frac{\alpha\Delta^2 d_v T}{2n\sigma^2} \ .$$

Here we adapted the proof of the formula: to obtain the right-handside, a value $\varepsilon^T_{w\to v}$ bigger than $\frac{\alpha\Delta^2}{2\sigma^2}$ was taken, so that the formula above is also true. Then, using the fact that node $v$ only sees its neighbors, we can use Lemma C.2 that allows us to take the mean conditionally on $v \cup \mathcal{N}(v)$ (for $\sigma^2 \geq \frac{\Delta^2\alpha(\alpha-1)}{2}$), leading to

$$\frac{1}{n}\left(\sum_{w\in\mathcal{N}(v)} \frac{\alpha\Delta^2}{2\sigma^2} + \sum_{w\in\mathcal{V}\setminus(\mathcal{N}(v)\cup\{v\})} \mathbb{E}\left[\varepsilon^T_{w\to v}(\alpha) \mid v \cup \mathcal{N}(v)\right]\right) \leq \frac{\alpha\Delta^2 d_v T}{n\sigma^2} \ .$$

In fact, we write it with the expected value, but all nodes are equal since node $v$ only sees its neighbors. Using the invariance under permutation of $\mathbb{E}\left[\varepsilon^T_{w\to v}(\alpha) v \cup \mathcal{N}(v)\right]$ over $w \in$

$\mathcal{V} \setminus (\mathcal{N}(v) \cup \{v\})$, we have that:

$$\frac{1}{n}\left(\sum_{w \in \mathcal{N}(v)} \frac{\alpha\Delta^2}{2\sigma^2} + (n - d_v)\varepsilon_{u\to v}^T\right) \leq \frac{\alpha\Delta^2 d_v T}{n\sigma^2}\,.$$

Rearranging this inequality gives the result. ■

## C.6 Differentially Private Decentralized Optimization

We consider Algorithm 6.3 with general time-varying matrices $W_t$. We assume that for all $t \geq 0$, $\lambda_{W_t} \geq \lambda$ for some fixed $\lambda > 0$. An instance of this setting is to sample different Erdös-Rényi random graphs at each communication round and adapt $W_t$ accordingly. Such graphs have a spectral gap that concentrates around 1, so that for $\lambda = 1/2$, with high probability $\lambda_{W_t} \geq \lambda$ will be verified [HKP19].

### C.6.1 Proof of Theorem 6.8 (Utility Analysis)

As before, we have a more general convergence result.

**Theorem C.7** (Utility analysis of Algorithm 6.3)**.** *Let* $K \geq \left\lceil \sqrt{\lambda}^{-1} \log\left(\max\left(n, \frac{\bar{\zeta}^2}{D\sigma^2 + \bar{\rho}^2}\right)\right)\right\rceil$. *For a suitable choice of step size parameters, the iterates* $(\theta^t)_{t\geq 0}$ *generated by Algorithm 6.3 verify:*

$$\mathbb{E}\left[\phi(\tilde{\theta}^T) - \phi(x^\star)\right] \leq \tilde{\mathcal{O}}\left(\frac{\bar{\rho}^2 + D\sigma^2}{n\mu T} + L\left\|\theta^0 - \theta^\star\right\|^2 e^{-\frac{T}{2\kappa}}\right)\,,$$

*where* $\tilde{\theta}^T = \frac{\sum_{t<T} \omega^t \bar{\theta}^t}{\sum_{t<T} \omega^t}$ *is a weighted average along the trajectory of the means* $\bar{\theta}^t = \frac{1}{n}\sum_{v\in\mathcal{V}} \theta_v^t$.

The proof of Theorem C.7 is a direct consequence of Theorem 2 in Koloskova et al. [Kol+20], and especially the formula in their Appendix A.4. We apply their result with $\bar{\rho}^2 + D\sigma^2$ instead of their $\bar{\sigma}^2$, $\tau = 1$ (no varying topology), and $p$ such that $1 - p = 2(1 - \sqrt{\lambda})^K \leq 2\min(\frac{1}{n}, \frac{D\sigma^2 + \bar{\rho}^2}{\bar{\zeta}^2})$.

### C.6.2 Proof of Theorem 6.9 (Privacy Analysis)

The function Lipschitzness can in fact be replaced by a more general assumption.

**Assumption C.8.** *We assume that, for some $\Delta_\phi^2 > 0$, for all $v$ in $\mathcal{V}$, and for all adjacent datasets $\mathcal{D} \sim_v \mathcal{D}'$ on $v$, we have:*

$$\sup_{\theta \in \mathbb{R}^D} \sup_{(x_v, x_v') \in \mathcal{D}_v \times \mathcal{D}_v'} \left\| \nabla_x \ell(\theta, x_v) - \nabla_x \ell(\theta, x_v') \right\|^2 \leq \Delta_\phi^2 \, .$$

**Theorem C.9** (Privacy analysis of Algorithm 6.3)**.** *Let $(W_t)_{0 \leq t < T}$ be a sequence of gossip matrices of spectral gap larger than $\lambda$. Let $u$ and $v$ be two distinct nodes in $\mathcal{V}$. After $T$ iterations of Algorithm 6.3 with $K \geq 1$, node $u$ is $(\alpha, \varepsilon_{u \to v}^T(\alpha))$-PNDP with respect to $v$, with:*

$$\varepsilon_{u \to v}^T(\alpha) \leq \frac{\Delta_\phi^2 \alpha}{2\sigma^2} \sum_{t=1}^{T} \sum_{j=t}^{T} \sum_{k=0}^{K-1} \sum_{w : \{v,w\} \in \mathcal{E}_j} \frac{(\Pi_{i=t}^{j-1} W_i^{K-1} W_j^k)_{u,w}^2}{\left\| (\Pi_{i=t}^{j-1} W_i^{K-1} W_j^k)_w \right\|^2} \, , \tag{C.4}$$

*Thus, for any $\varepsilon > 0$, Algorithm 6.3 with $T^{\mathrm{stop}}(\kappa, \sigma^2, n)$ steps and for $K$ as in Theorem 6.8, there exists $f$ such that the algorithm is $(\alpha, f)$-pairwise network DP, with:*

$$\forall v \in \mathcal{V} \, , \quad \bar{\varepsilon}_v \leq \varepsilon \quad \textit{and} \quad \mathbb{E}\left[ \phi(\tilde{\theta}^{\mathrm{out}}) - \phi(\theta^\star) \right] \leq \tilde{\mathcal{O}}\left( \frac{\alpha D \Delta_\phi^2 \bar{d}}{n^2 \mu \varepsilon \sqrt{\lambda}} + \frac{\bar{\rho}^2}{nL} \right) \, ,$$

*where $\bar{d} = \sup_{v \in \mathcal{V}} \bar{d}_v$ for $\bar{d}_v = \frac{1}{T} \sum_{t<T} |\{w \in \mathcal{V} : \{v, w\} \in \mathcal{E}_t\}|$ the mean degree of node $v$ throughout time.*

*Proof of Theorem C.9.* The information leaked by $u$ to $v$ up to iteration $T$ of Algorithm 6.3 consists in the $T$ (stochastic) gradients locally computed at node $u$ and gossiped through the graph, using the *Muffliato* algorithm. Note that the gradient used at iteration $t$ continues to leak information to some nodes after the $K$ gossip averaging steps of iteration $t$, as the information continues to flow in the graph until the end of the *Muffliato*-GD/SGD algorithm, i.e. for the subsequent $(T - t)$ iterations.

For the last round however, using Theorem 6.1, the privacy loss is given by that of *Muffliato*:

$$\frac{\Delta_\phi^2 \alpha}{2\sigma^2} \sum_{k=0}^{K-1} \sum_{w : \{v,w\} \in \mathcal{E}_T} \frac{(W_T^k)_{u,w}^2}{\left\| (W_T^k)_w \right\|^2} \, .$$

Then, the total privacy loss of a given round $t$ can be upper bounded by the privacy loss of running *Muffliato* for $(T - t)K$ steps. Thus, we sum each of the round $t$ over the $(T - t)K$

subsequent steps, leading to the formula:

$$\frac{\Delta_\phi^2 \alpha}{2\sigma^2} \sum_{j=t}^{T} \sum_{k=0}^{K-1} \sum_{w:\{v,w\}\in\mathcal{E}_j} \frac{\left((\Pi_{i=t}^{j-1} W_i^{K-1}) W_j^k\right)_{u,w}^2}{\left\| \left((\Pi_{i=t}^{j-1} W_i^{K-1}) W_j^k\right)_w \right\|^2} ,$$

where $\mathcal{E}_t$ are the edges of the graph drawn at time $t$. We obtain the upper bound on $\varepsilon_{u\to v}^T(\alpha)$ stated in the first part of Theorem C.9 by summing this expression over $t < T$, and the simplified version in the first part of the Theorem 6.9 follows directly from the fact that we consider the graph fixed across iterations.

For the second inequality, we have, by summing:

$$\bar{\varepsilon}_v = \frac{1}{n} \sum_{u\neq v} \varepsilon_{u\to v}^T(\alpha) \leq \frac{KT^2 \bar{d}_v \Delta_\phi^2 \alpha}{2n\sigma^2} \leq \frac{KT^2 \bar{d} \Delta_\phi^2 \alpha}{2n\sigma^2} .$$

In order to reach a precision $\frac{D\sigma^2+\tilde{\rho}^2}{n}$, $T = \mathcal{O}(\kappa \log(D\sigma^2/n)) = T^{\text{stop}}(\kappa, \sigma^2, n)$ iterations are required. Using $K = \tilde{\mathcal{O}}(1/\sqrt{\lambda})$, we have:

$$\bar{\varepsilon}_v = \mathcal{O}\left( \frac{\bar{d}\Delta_\phi^2 \alpha}{2n\sigma^2} \kappa^2 \sqrt{\lambda}^{-1} \log^2(D\sigma^2/n) \right) .$$

Taking $\sigma^2$ such that $\frac{\bar{d}\Delta_\phi^2 \alpha}{2n\sigma^2} \kappa^2 \sqrt{\lambda}^{-1} \log^2(\sigma^2/n) = \varepsilon$ and plugging into Theorem 6.8 yields the desired result. ■

## C.7 Extensions to Collusion and Group Privacy

In this section, we discuss natural extensions of our privacy definitions to the case of colluding nodes, and to group privacy.

### C.7.1 Presence of Colluding Nodes

#### Definitions

The notions of pairwise network DP we introduced in Equation (6.3) can naturally be extended to account for potential collusions. For $V \subset \mathcal{V}$ a set of colluding nodes, we define the *view* of the colluders as:

$$\mathcal{O}_V\left(\mathcal{A}(\mathcal{D})\right) = \bigcup_{v\in V} \mathcal{O}_V\left(\mathcal{A}(\mathcal{D})\right) ,$$

or equivalently as:

$$\mathcal{O}_V\left(\mathcal{A}(\mathcal{D})\right) = \{(u, m(t), v) \in \mathcal{A}(\mathcal{D}) \quad \text{such that} \quad \{u, v\} \in \mathcal{E}\,, v \in V\}\,.$$

Below, $\mathcal{P}(\mathcal{V})$ denotes the powerset of $\mathcal{V}$.

**Definition C.10** (Pairwise Network DP with colluders)**.** *For $f : \mathcal{V} \times \mathcal{P}(\mathcal{V}) \to \mathbb{R}^+$, an algorithm $\mathcal{A}$ satisfies $(\alpha, f)$-pairwise network DP if for all users $u \in \mathcal{V}$, pairs of neighboring datasets $D \sim_u D'$, and any potential set of colluders $V \in \mathcal{V}$ such that $u \notin V$, we have:*

$$D_\alpha(\mathcal{O}_V(\mathcal{A}(\mathcal{D})) \,||\, \mathcal{O}_V(\mathcal{A}(\mathcal{D}'))) \leq f(u, V)\,. \tag{C.5}$$

*We note $f(u, V) = \varepsilon_{u\to V}$ the privacy leaked to the colluding nodes $V$ from $u$ and say that $u$ is $(\alpha, \varepsilon_{u\to V})$-PNDP with respect to $V$ if only inequality (C.5) holds for $f(u, V)$. Finally, if for a function $f : \mathcal{V} \times \mathcal{P}(\mathcal{V}) \to \mathbb{R}$, inequality (C.5) holds for all $(u, V) \in \mathcal{V} \times \mathcal{P}(\mathcal{V})$ such that $u \notin V$, we say that $\mathcal{A}$ is $(\alpha, f)$-pairwise NDP.*

This definition quantifies the privacy loss of a node $u$ with respect to the collusion of any possible subset $V$ of nodes, and thus generalizes the definition of the main text (which corresponds to restricting $V$ such that $|V| = 1$).

The proofs of this section are actually direct consequences of the proof techniques of our results without colluders, by replacing $\mathcal{O}_v$ (the view of a colluder) by $\mathcal{O}_V$ (the view of the colluding set). Roughly speaking, $V$ can be seen as a unique abstract node, resulting from the fusion of all its nodes.

### Adapting Theorem 6.1 and the Resulting Corollaries

For $w \in \mathcal{V}$ and $V \in \mathcal{P}(\mathcal{V})$, let $\mathcal{P}^t_{\{V,w\}} = \{s < t \,:\, \exists v \in V\,, \{v, w\} \in \mathcal{E}_s\}$ the times (up to time $t$) at which an edge $\{v, w\}$ for any $v \in V$ is activated *i.e.* the times at which there is a communication between $w$ and a colluder. For $t \geq 0$ and $V \subset \mathcal{V}$, let $\mathcal{N}_t(V)$ be the neighbors in $G_t$ of the colluders set $V$, defined as:

$$\mathcal{N}_t(V) = \{w \in \mathcal{V} \setminus V \,|\, \exists v \in V\,, \{v, w\} \in \mathcal{E}_t\}\,.$$

For $T \geq 1$, $\sum_{t<T} |\mathcal{N}_t(V)|$ is thus the total number of communications in which colluders are involved with.

**Theorem C.11.** *Let $T \geq 1$, $u \in \mathcal{V}$ and $V \subset \mathcal{V}$ such that $u \notin V$, and $\alpha > 1$. Then the algorithm $\mathcal{A}^T$ is $(\alpha, f)$-PNDP with::*

$$f(u,V) \leq \frac{\alpha\Delta^2}{2\sigma^2} \sum_{w\in\mathcal{V}} \sum_{t\in\mathcal{P}^T_{\{V,w\}}} \frac{(W_{0:t})^2_{u,w}}{\|(W_{0:t})_w\|^2}\,. \tag{C.6}$$

*Consequently, there exists $f$ such that $\mathcal{A}^T$ is $(\alpha, f)$-PNDP with:*

$$\bar{\varepsilon}_V = \frac{1}{n}\sum_{u\in\mathcal{V}\setminus V} f(u,V) \leq \frac{\alpha\Delta^2}{2n\sigma^2} \sum_{t<T} |\mathcal{N}_t(V)|\,, \tag{C.7}$$

*where $\sum_{t<T}|\mathcal{N}_t(V)|$ is the total number of communications a colluding set $V$ is involved with, up to time $T$.*

We now consider the synchronous *Muffliato* algorithm (Algorithm 6.1) with colluders.

**Corollary C.12.** *Let $u \in \mathcal{V}$ and $V \in \mathcal{P}(\mathcal{V})$ such that $u \notin V$, $\alpha > 0$. After $T$ iterations of Algorithm 6.1, node $u$ is $(\alpha, \varepsilon^T_{u\to V}(\alpha))$-PNDP with respect to $V$, with:*

$$\varepsilon^T_{u\to V}(\alpha) \leq \frac{\alpha n}{2\sigma^2} \max_{w\sim V} W^{-2}_{v,w} \sum_{t=1}^{T} \mathbb{P}\left(X^t \in V | X^0 = u\right)^2\,,$$

*where $(X_t)_t$ is the random walk on graph $G$, with transitions $W$, and $w \sim V$ if $w \notin V$ and if there exists $v \in \mathcal{V}$ such that $\{v,w\} \in \mathcal{E}$.*

**Corollary C.13.** *There exists $f : \mathcal{V} \times \mathcal{P}(\mathcal{V}) \to \mathbb{R}^+$ such that Algorithm 6.1 after $T$ steps is $(\alpha, f)$-PNDP with the following privacy-utility guarantees for any $V \subset \mathcal{V}$:*

$$\bar{\varepsilon}_V = \frac{1}{n}\sum_{u\in\mathcal{V}\setminus V} f(u,V) \leq \varepsilon \quad , \qquad \frac{1}{2n}\sum_{v\in\mathcal{V}} \mathbb{E}\left[\left\|x^{\text{out}}_v - \overline{x}\right\|^2\right] \leq \tilde{\mathcal{O}}\left(\alpha \frac{d_V\Delta^2}{\varepsilon n^2\sqrt{\lambda_W}}\right)\,,$$

*where $x^{\text{out}}$ is the output of Algorithm 6.1 after $T^{\text{stop}}(x, W, \sigma^2)$ steps for $\sigma^2 = \frac{d_V\Delta^2}{2\alpha\varepsilon}$, and $\tilde{\mathcal{O}}$ hides logarithmic factors in $n$ and $\varepsilon$. $d_V$ is the degree of set $V$, defined as the number of $w \in \mathcal{V}\setminus V$ such that there exists $v \in V$, $\{v,w\} \in \mathcal{E}$.*

**Corollary C.14** (*Muffliato* on a random graph with collusions)**.** *Let $\alpha > 1$, $T \geq 0$, $\sigma^2 \geq \frac{\Delta^2\alpha(\alpha-1)}{2}$ and $q = c\frac{\log(n)}{n}$ for $c > 1$. Let $u \in \mathcal{V}$ and $V \in \mathcal{P}(\mathcal{V})$ such that $u \notin V$. After running Algorithm 6.1 on an Erdos Rényi random graph of parameters $(n, q)$ and under the assumptions of*

*Theorem 6.6, node $u$ is $(\alpha, \varepsilon^T_{u\to v}(\alpha))$-PNDP with respect to colluders $V$, with:*

$$\varepsilon^T_{u\to V}(\alpha) \leq \begin{cases} \dfrac{\alpha}{2\sigma^2} & \text{with probability } 1-(1-q)^{|V|} \\ \dfrac{\alpha}{\sigma^2}\dfrac{Td_V}{n-d_V} & \text{with probability } (1-q)^{|V|} \end{cases}.$$

### Compensating for Collusions with Time-Varying Graph Sampling

We now consider the decentralized optimization problem of Section 6.3 in the presence of colluders, and analyze its privacy with time-varying graph sampling as in Appendix C.6.2.

The motivation for this is that, if the graph is fixed, node $u$ will suffer from poor privacy guarantees (the same as in LDP) with respect to the colluding set $V$ as soon as $u$ is in $\mathcal{N}(V) = \mathcal{N}_0(V)$ (i.e., one of colluders in $V$ is a neighbor of $u$). Even if the graph is sampled randomly, this will happen with probability that increases with $|V|$. In contrast, for time-varying random graphs sampled independently at each communication round and for sufficiently many communication rounds (i.e., large enough condition number $\kappa$), it becomes unlikely that $u$ is in $\mathcal{N}_t(V)$ for many rounds $t$, and therefore the privacy guarantees with respect to $V$ can improve.

Below, we consider Algorithm 6.3 with time-varying graphs (and associated gossip matrices $(W_t)_{t\geq 0}$) sampled in an *i.i.d.* fashion at each communication round as Erdös-Rényi graphs of parameters $n, q = \frac{c\log(n)}{n}$ for some $c > 1$, such that they verify $\lambda_{W_t} \geq \lambda > 0$ for all $t$ (as noted before, this happens with high probability for $\lambda$ of order 1 [HKP19]). In this context, we have the following result.

**Proposition C.15.** *Let $\alpha > 1$, $T \geq 0$, $\sigma^2 \geq \frac{\Delta^2\alpha(\alpha-1)}{2}$. Let $u \in \mathcal{V}$ and $V \in \mathcal{P}(\mathcal{V})$ such that $u \notin V$. After running Algorithm 6.3 under the assumptions described above and the function assumptions of Theorem 6.9, node $u$ is $(\alpha, \varepsilon^T_{u\to v}(\alpha))$-PNDP with respect to colluders $V$, with:*

$$\varepsilon^T_{u\to V}(\alpha) \leq \frac{\Delta^2_\phi \alpha}{\sigma^2} \sum_{t=0}^{T^{\text{stop}}} \beta_t + (1-\beta_t)\frac{K|\mathcal{N}_t(V)|}{n-|\mathcal{N}_t(V)|},$$

*where $T^{\text{stop}} = \tilde{\mathcal{O}}(\kappa)$ and $K = \tilde{\mathcal{O}}(1/\sqrt{\lambda})$ (see Theorem 6.8), $(\beta_t)_t$ are* i.i.d. *Bernoulli random variables of parameter $\mathbb{P}(\exists v \in \mathcal{V}, \{u,v\} \in \mathcal{E}_t) = 1-(1-q)^{|V|}$, and $|\mathcal{N}_t(V)|$ is the number of neighbors of $V$ in the graph sampled at iteration $t$, of order $\frac{c|V|\log(n)}{n}$.*

### Discussion

Generally speaking, our bounds degrade in presence of colluding nodes. This is a fundamental limitation of our approach that considers only privacy amplification due to decentralization. By definition, our privacy guarantees can only provide amplification as long as the view of

the attackers is smaller than the one of the omniscient attacker considered in local differential privacy, i.e $\mathcal{O}_V(\mathcal{A}^T) \subsetneq \mathcal{A}^T$. A condition for having equality corresponds to observing all messages that are transmitted. In the case of a fixed graph, this can be characterized by the fact that $V$ contains a dominating set for the graph. For Erdos Rényi graphs or exponential graphs, there exists dominating sets of size $\mathcal{O}(\log n)$, thus it is meaningless to expect guarantees for all possible sets of colluding nodes of that size. However, if some/most colluding nodes are actually far from the target node $u$ in the graph, then good privacy amplification can still be achieved. This can be precisely measured by Equation C.6.

### C.7.2 Group Privacy

Symmetrically to the problem of collusions, where there are several attackers, one can study group privacy, where privacy guarantees are computed towards a group rather than a single individual. This is useful when some users have correlated data (e.g., close family members). The usual notion of group privacy [see e.g., DR14] would provide a guarantee against all groups of users of a given size. However, this standard definition would provide pessimistic guarantees in some cases as it does not take advantage of the fact that groups may correspond to specific subgraphs. Hence, we propose the following definition.

**Definition C.16** (Group Pairwise Network DP)**.** *For $f : \mathcal{V} \times \mathcal{P}(\mathcal{V}) \to \mathbb{R}^+$, an algorithm $\mathcal{A}$ satisfies $(\alpha, f)$-group pairwise network DP if for all set of users $U \subset \mathcal{V}$, pairs of neighboring datasets $D \sim_U D'$, and any vertex $v \in \mathcal{V}$, we have:*

$$D_\alpha(\mathcal{O}_v(\mathcal{A}(\mathcal{D})) \,||\, \mathcal{O}_v(\mathcal{A}(\mathcal{D}'))) \leq f(U, v)\,. \tag{C.8}$$

The modifications of the theorems are similar to the case of collusion, summing over the nodes in $U$ (instead of summing over the nodes in $V$ for the case of collusion). The following theorem gives the general case corresponding to Theorem 6.1. The other results can be adapted in the same way.

**Theorem C.17.** *Let $T \geq 1$ and denote by $\mathcal{P}^T_{\{v,w\}} = \{s < T \,:\, \{v, w\} \in \mathcal{E}_s\}$ the set of time-steps with communication along edge $\{v, w\}$. With $\Delta$ the sensitivity, $\mathcal{A}^T$ is $(\alpha, f)$-group PNDP for with:*

$$f(U, v) = \frac{\alpha\Delta^2}{2\sigma^2} \sum_{w \in \mathcal{V}} \sum_{t \in \mathcal{P}^T_{\{v,w\}}} \frac{\sum_{u \in U} (W_{0:t})^2_{u,w}}{\|(W_{0:t})_w\|^2}\,. \tag{C.9}$$

Note that in some configurations, this bound bound is clearly sub-optimal, as summing does not take into account cases where the information gathered by some of the nodes in the group can be seen as a post-processing of information received by other nodes in the group. In

such cases, analyzing group privacy by considering a modified graph where the nodes in the group are merged into a single node, with edge/weights adjusted accordingly, would yield better results.

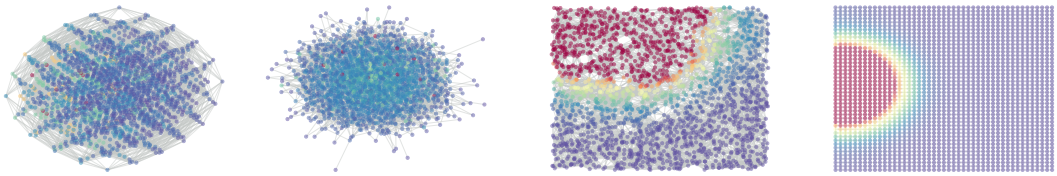


**Figure C.1** – Level of the privacy loss for each node (color) with respect to a fixed node in the graph. These graphs corresponds to the graphs used in Figure 6.1(a): from left to right, exponential graph, Erdos-Rényi graph, geometric random graph and grid.

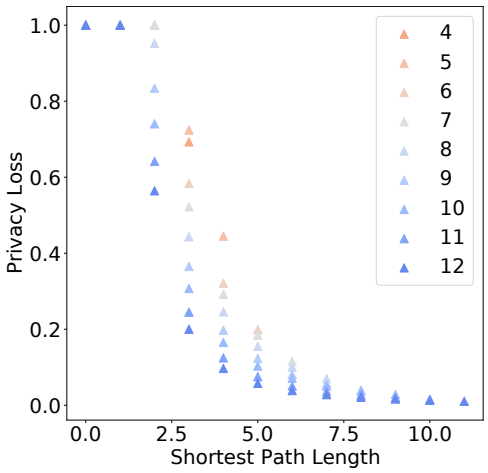



**Figure C.2** – Privacy loss for the exponential graph with respect to the number of nodes $n$ (following powers of 2).

## C.8 Additional Numerical Experiments

### C.8.1 Extra Synthetic graphs

Figure 6.1(a) summarizes the result of *Muffliato* according to the shortest path length. However, other characteristics of the topology can play a role in the privacy leakage. Thus, we show the graph representation for each of the synthetic graphs we considered in Figure C.1.

We also report in Figure C.2 how privacy guarantees improve when $n$ increases for the exponential graph. We see that privacy guarantees improve with $n$: distance between nodes increases, but also the number of nodes with whom the contribution of a specific node is mixed. This is especially significant for pairs of nodes that are not direct neighbors but at short distance of each other.

### C.8.2 Proof of Fixed Privacy Loss for Exponential Graphs

For an exponential graph, the pairwise privacy loss is fully determined by the shortest length path, i.e $f(u,v) = g(d(u,v))$ where $d : \mathcal{V} \to \mathbb{N}$ is the function that returns the length of the shortest path between $u$ and $v$.

This result is a consequence of the invariance per vertex permutation in the hypercube. For the hypercube with $2^m$ vertices, each vertex can be represented by a $m$-tuple in $\{0,1\}$, where there is an edge if and only if two vertices share all values of their tuple but one. Let us now fix two pairs of vertices $(u,v)$ and $(u',v')$ with the same distance between them. To prove that their privacy loss is the same, it is sufficient to exhibit an graph isomorphism $\Phi$ that sends $(u,v)$ on $(u',v')$.

By construction, $d(u,v)$ corresponds to the number of coordinates that differ between their tuple, and the same holds for $(u',v')$. The set of equal coordinates $Fix(u,v)$ is thus of the same size $m-d(u,v)$ than $Fix(u',v')$. Hence we can construct a bijective function $b$ of the coordinates that is stable for the set of fixed coordinates

$$b(Fix(u,v)) = Fix(u',v') \quad b(\{1,2,\dots,m\} \setminus Fix(u,v)) = \{1,2,\dots,m\} \setminus Fix(u',v')$$

Finally, noting that $0$ and $1$ play the same role, we match each coordinate accordingly to the value defined by our couple. We thus define our isomorphism per coordinate $\Phi(w) = (\Phi_1(w),\dots,\Phi_m(w))$ with $\Phi_i(w) = s(w_{b^{-1}(i)})$ where $s$ is the identity function if $u_{b^{-1}(i)} = u_i$ and the swap function otherwise. This function is clearly an isomorphism: by construction it is a bijection, and the edges still exist if and only if the vertices differ on only one coordinate. We have $\Phi(u) = u'$ and $\Phi(v) = v'$ and thus the privacy loss is equal between the two pairs of vertices.

### C.8.3 Random Geometric Graphs

Geometric graphs are examples of possible use cases of Pairwise Network Differential Privacy. Constructing edges when nodes are at a distance below a given threshold naturally models short-range wireless communications such as Bluetooth. In this situation, the Euclidean distance between nodes is a convenient indicator for setting the privacy loss. Indeed, it is a parameter that we can measure, and it can match the users expectations in terms of privacy loss. For instance, if direct neighbors in the graph correspond to people within $5$ meters around the sender, some information are bound to be available to them independently of what may be revealed by the communication itself: sensitive attributes such a gender, age, or overall physical fitness are leaked simply from physical proximity. However, the user might have stronger privacy expectations for people far away. Hence, having privacy guarantees as function of the Euclidean distance can be particularly interesting.

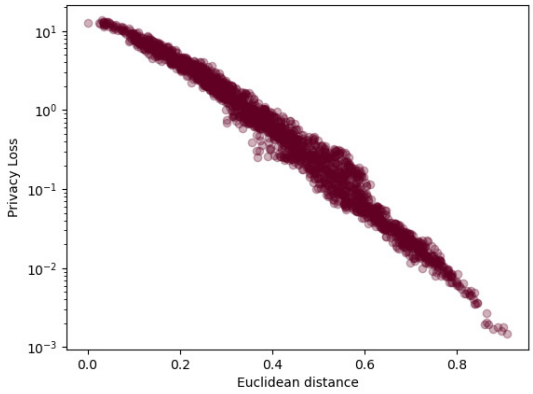



**Figure C.3** – Privacy towards all the nodes as function of the Euclidean distance in a random geometric graph. We see a high level of correlation between the Euclidean distance and the privacy loss.

**Table C.1** – Parameter for the logistic regression

| **Parameters** | Value |
|---|---|
| # of trials | 10 |
| Step-size | 0.7 |
| # of nodes | 2000 or 4000 |
| probability of edges $q$ | $\log(n)/n$ |
| score | Mean accuracy |

Our experiments show that the privacy loss is extremely well correlated to the Euclidean distance, as represented in Figure C.3. It is thus possible to design algorithms where one could impose Pairwise Network DP for a function $f(u, v) = g(\|z_u - z_v\|)$ where $g$ is a non-increasing function and $z_u$ and $z_v$ are the geolocation of nodes $u$ and $v$.

### C.8.4 Facebook Ego Graphs

We report figure on the other nine graphs of the Facebook Ego dataset, following the same methodology and scale. Across these graphs, we can see that privacy losses depending on visible communities is consistent through datasets, and become more consistent as the number of nodes increase.

### C.8.5 Logistic Regression on Houses Dataset

We report in Table C.1 the parameters used in the experiments of Figure 6.1(c).

**Figure C.4** – Privacy loss on the 9 other Facebook Ego graphs, following the same methodology as in Figure 6.1(b).

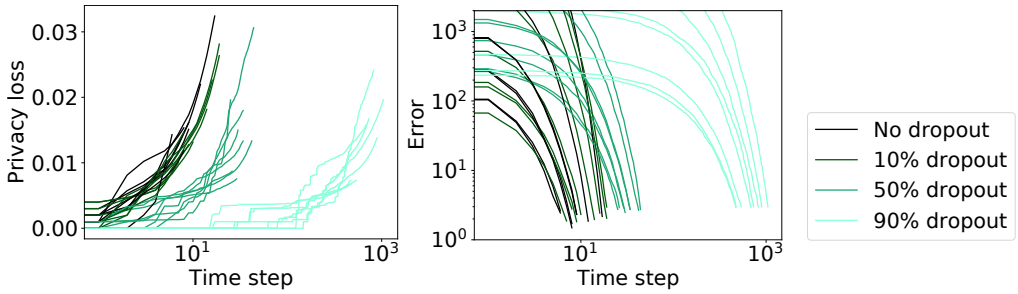



**Figure C.5** – Simulation (10 runs) of a dropout scenario (with different levels of dropout) with $n = 1000$, where at each gossip step an Erdös-Rényi graph with parameter $p = 0.002$ over the set of available users. (a) Left: Privacy loss of a node across iterations ;(b) Right: Convergence of the same runs to the mean.

**Table C.2** – Total Privacy loss in function of dropout

| **Dropout** | Privacy loss |
|---|---|
| No dropout | $(1.7 \pm 0.6) \times 10^{-2}$ |
| 10% dropout | $(1.8 \pm 0.6) \times 10^{-2}$ |
| 50% dropout | $(1.5 \pm 0.7) \times 10^{-2}$ |
| 90% dropout | $(1.4 \pm 0.6) \times 10^{-2}$ |

### C.8.6 Modeling User Dropout using Time-Varying Graphs

In Theorem 6.1, we analyzed the very generic case where the gossip matrix is arbitrary at each time-step. Then, for deriving the convergence rate and obtaining closed-form privacy-utility trade-offs and using acceleration, we focused on fixed gossip matrices until the convergence of each gossip averaging step. In this subsection, we show empirically that we can still reach a good privacy-utility trade-offs when the gossip matrix change at each communication.

In particular, our experiment focuses on modelling user dropouts. Specifically, at each time step, the availability of each node is modeled by an independent Bernouilli random variable and we draw a new Erdos Rényi graph over the set of available nodes. We assume that at each step, each node has the same given probability to be active so as to ensure convergence to the mean of the values. One could design more sophisticated dropout models, as long as the contributions of nodes remain balanced.

We vary the expected level of available nodes at each step from 10 to 90%. Note that the number of iterations needed for convergence becomes stochastic: we thus set an arbitrary number of iterations experimentally, and average several runs. For simplicity, we do not perform gossip acceleration. We report several runs at each dropout level in Figure C.5(a), and report the privacy loss and its standard deviation at each dropout level in Table C.2. As expected, the convergence time increases with the proportion of inactive users, but the

achievable privacy-utility trade-off is not significantly impacted. Therefore, our approach scales gracefully difficulty to the situations where there is dropout.

## C.9 Broader Impact Assessment

Our work promotes increased privacy in federated ML. The potential longer-term benefits of our work in this respect are a wider adoption of privacy-preserving and fully decentralized ML solutions by service providers thanks to the improved privacy-utility guarantees, as well as better confidence of users and the general public in the ability of decentralized ML systems to avoid catastrophic data leaks. In particular, our work shows that the advantages of decentralized methods over centralized ones have been overlooked, due to the lack of privacy analysis able to capture the benefits of decentralized algorithms.

Conversely, there are potential risks of accidental or deliberate misuse of our work in the sense that it could give a false sense of privacy to users if weak privacy parameters are used in deployment. This applies to all work on differential privacy. More applied research is needed towards developing a methodology to choose appropriate privacy parameters in a data-driven manner and to reliably assess the provided protection in practical use-cases.

Our work specifically proposes a varying privacy budget that depends on the relation between users, which might be misused for giving smaller privacy guarantees than the ones that would be designed otherwise. Modularity in privacy guarantees has however already been studied, for instance in [Cha+13] where the privacy budget is a function of a metric on the input space. Informally, defining what is an acceptable privacy budget based on the context in which some information is revealed is in line with the idea of contextual integrity [BGN17]. According to Helen Nissenbaum's theory, the privacy expectations should take into account five elements: the sender, the receiver, the message, the medium of transmission and the purpose. Mathematically, adapting the privacy guarantee to the receiver and promoting peer-to-peer communications for building a global model thus naturally fits this view. In particular, contextual integrity emphasizes that the privacy budget should not only depend on the information being transmitted, but also on who receives it.

# Appendix D

# Proofs and Additional Results for Chapter 7

## D.1 Proof of Theorem 7.1 (Real Summation on the Complete Graph)

*Proof.* We will prove an $(\varepsilon_f, \delta_f)$-DP guarantee for Algorithm 7.1. We note that our proof does not require the global time counter $t$ to be hidden from users (i.e., the result holds even if users receiving the token know how many users have added contributions to the token since its last visit).

Let us fix two distinct users $u$ and $v$. We aim to quantify how much information about the private data of user $u$ is leaked to $v$ from the visits of the token. Recall that we assume the token path to be hidden, including the previous sender and the next receiver. We can thus define the view $\mathcal{O}_v$ of user $v$ by:

$$\mathcal{O}_v(\mathcal{A}(D)) = (\tau_{k_i})_{i=1}^{T_v}, \tag{D.1}$$

where $k_i$ is the $i$-th time that $v$ receives the token, $\tau_{k_i}$ the corresponding value of the token, and $T_v$ the number of times that $v$ had the token during the whole execution of the protocol.

We aim at bounding the privacy loss with respect to the contributions of $u$ from the point of view of $v$. We call "cycle" the portion of the walk between two visits of the token to $v$. We first note that we can decompose the walk in cycles by cutting the walk at each $k_i$. If a contribution of $u$ happens at time $t$, there is single $i$ such a $k_i < t < k_{i+1}$.[1] Note that the token values observed before $t$ do not depend on the contribution of $u$ at time $t$. Moreover, it is sufficient to bound the privacy loss induced by the observation of the token at $k_{i+1}$: indeed, by the post-processing property of DP, no additional privacy loss with respect to $v$ will occur for observations posterior to $k_{i+1}$.

To allow the use of privacy amplification by subsampling results [BBG18], we will actually consider a variant of the actual walk. We assume that if $n$ steps have occurred since the last visit of the token to $v$, the value of the token at that time is observed by $v$ "for free". As the information leaked to v by the actual walk can be obtained by post-processing of this *fictive* walk, it is sufficient to prove privacy guarantees on the fictive walk.

The number of observations of the token by $v$ can be bounded by the "real" observations (from actual visits of the token) plus the fictive ones. By definition, there is no more than $T/n$ fictive observations of the token. We now bound the number of real visits of the token to $v$.

As the user receiving the token at a given step is chosen uniformly at random and independently from the other steps, there is a probability of $1/n$ that the token is at $v$ at any given step. Thus, the number of visits $T_v$ to $v$ follows a binomial law $\mathcal{B}(T, 1/n)$. We bound it by $N_v$ with probability $1 - \hat{\delta}$ using Chernoff. Recall that the Chernoff bound allows to upper bound (with

[1] If the contribution of $u$ occurs before the first passage of the token at $v$, we can take $k_i = 0$. As for contributions occurring *after* the last passage of the token at $v$, they do not incur any privacy loss.

high probability) the sum of independent random variables $X_1, \ldots, X_T$ of expected value $p$, for any real $\alpha \in [0, 1]$:

$$\mathbb{P}\left(\sum_{i=1}^{T} X_i \geq (1+\alpha)Tp\right) \leq e^{-\alpha^2 pT/3}.$$

In our case, we want to upper bound the probability that the number of contributions $T_v$ of a given user $v$ exceeds some threshold $N_v$ by $\hat{\delta}$. Using the previous bound for $p = 1/n$ and $\alpha = \sqrt{\frac{3n\log(1/\hat{\delta})}{T}}$, by considering the random variables equal to 1 it $v$ has the token and 0 otherwise, we have:

$$\mathbb{P}\Bigg(T_v \geq \underbrace{\frac{T}{n} + \sqrt{\frac{3T}{n}\log(1/\hat{\delta})}}_{N_v}\Bigg) \leq \hat{\delta}.$$

Let us now upper bound the privacy loss that occurs during a fixed cycle. The information revealed to $v$ by a cycle of size $1 \leq m \leq n$ can be seen as a mechanism $\mathcal{M} = \mathcal{A} \circ \mathcal{S}$, where $\mathcal{A}$ corresponds to the aggregation of $m$ values with $m$ additions of Gaussian noise, and $\mathcal{S}$ corresponds to subsampling with replacement $m$ users among $n$ (as each user is uniformly chosen at random at each step). The base mechanism $\mathcal{A}$ satisfies $(\varepsilon/\sqrt{m}, \delta)$-DP.

According to Theorem 10 from [BBG18]: given $n$ users and $m$ the size of the cycle, the privacy of $\mathcal{M} = \mathcal{A} \circ \mathcal{S}$ satisfies $(\varepsilon_{cycle}, \delta_{cycle})$ with:

$$\varepsilon_{cycle} = \log(1 + (1 - (1 - 1/n)^m)(e^{\varepsilon_{\mathcal{A}}} - 1)), \tag{D.2}$$

Hence, for a cycle of size $m$, $\mathcal{M}$ satisfies $(\varepsilon_{cycle}, \delta_{cycle})$-DP with

$$\varepsilon_{cycle} \leq \log\left(1 + (1 - \left(1 - \frac{1}{n}\right)^m)(e^{\varepsilon/\sqrt{m}} - 1)\right).$$

Using the fact that $\varepsilon \leq 1$, we can upper bound $e^{\varepsilon/\sqrt{m}} - 1$ by $2\varepsilon/\sqrt{m}$. Moreover, as $1/n < 0.58$, we have $-\frac{3}{2n} \leq \log(1 - 1/n)$. So we have

$$1 - \exp(m\log(1 - 1/n)) \leq 1 - \exp\left(-\frac{3m}{2n}\right) \leq \frac{3m}{2n}.$$

Combining the two upper bounds and the classical inequality $\log(1 + x) \leq x$ gives us:

$$\varepsilon_{cycle} \leq \frac{3\sqrt{m}\varepsilon}{n} \leq \frac{3\varepsilon}{\sqrt{n}}.$$

Hence we can upper bound the privacy loss of each cycle by $\frac{3\varepsilon}{\sqrt{n}}$ regardless of its length $m$. Finally, we use advanced composition to account for the privacy losses of all $T/n + N_v$ cycles,

leading to the following bound:

$$\varepsilon_f \leq \sqrt{\left(\frac{4T}{n} + 2\sqrt{\frac{3T}{n}\log(1/\hat{\delta})}\right)\log(1/\delta')}\frac{3\varepsilon}{\sqrt{n}} + \sqrt{\frac{2T}{n} + \sqrt{\frac{3T}{n}\log(1/\hat{\delta})}}\varepsilon(e^{3\varepsilon/\sqrt{n)}} - 1),$$

with $\delta_f = (N_v + T/n)\delta_{cycle} + \delta' + \hat{\delta}$. ■

### D.1.1 Precise $\delta_{cycle}$ computation

In the previous proof, we used $\delta_{cycle}$ to derive our privacy guarantee as defined in [BBG18]. We detail here how the term can be computed and discuss its value. For running a mechanism $\mathcal{A}$ on a sample of $m$ samples drawn with replacement from a set of size $n$, the bound given by [BBG18] is:

$$\delta_{cycle} = \sum_{k=1}^{m} \binom{m}{k} \left(\frac{1}{n}\right)^k \left(1 - \frac{1}{n}\right)^{m-k} \delta_{\mathcal{A},k}$$

In this equation, the term $\delta_{\mathcal{A},k}$ does not admit a direct closed form in most cases. Its definition is the following:

$$\delta_{\mathcal{A},k}(\varepsilon) = \sup_{d(x,x')\leq k} D^h_{e^\varepsilon}\left(\mathcal{A}(x) \| \mathcal{A}\left(x'\right)\right)$$

which corresponds to the guarantee that holds for group privacy of size $k$. $D^h$ is the hockey-stick divergence (see Definition 3.3). By applying the definition of group privacy and composing $k$ times the inequality constraints, one can derive the following formula:

$$\delta_{\mathcal{A},k}(\varepsilon) \leq \left(e^\varepsilon - 1\right)\delta_{\mathcal{A}}(\varepsilon/k)/\left(e^{\varepsilon/k} - 1\right)$$

The term makes appears $\delta_{\mathcal{A}}(\varepsilon/k)$ which has no explicit connection with $\delta_{\mathcal{A}}(\varepsilon)$ in general, and thus does not admit a closed form formula either. However, in the case of Gaussian mechanism, it is possible to use the following formula:

$$\delta_{\mathcal{A}}\left(\varepsilon\right) = \Phi(\theta/2 - \varepsilon/\theta) - e^\varepsilon \Phi(-\theta/2 - \varepsilon/\theta)$$

with $\theta = \delta/\sigma$ and $\Phi$ the CDF of the standard normal distribution, which allows to compute numerically the resulting value of $\delta_{cycle}$.

In particular, for all values used for the numerical experiments (Section 7.2.4), applying this method of computation gives $\delta \leq \delta_{cycle}$, and thus we use $\delta$ as a simplification. It seems

**Algorithm D.1:** Private histogram computation on a complete graph

Initialize $\tau \in \mathbb{N}^L$;
**for** $t = 1$ **to** $T$ **do**
  Draw $u \sim \mathcal{U}(1, \dots, n)$;
  $y_u^k \leftarrow RR_\gamma(x_u^k)$;
  $\tau[y_u^k] \leftarrow \tau[y_u^k] + 1$;
**for** $i = 0$ **to** $L - 1$ **do**
  $\tau[i] \leftarrow \frac{\tau[i] - \gamma/L}{1-\gamma}$;
**return** $\tau$

that this inequality holds experimentally for reasonable values of $\sigma$ and $\varepsilon$. Two facts can give intuition on why $\delta_{cycle} \leq \delta$ numerically.

Firstly, note that for other sampling schemes, such as sampling without replacement and Poisson sampling, the corresponding $\delta_{sampling}$ is provably smaller than the $\delta$ of the base mechanism. Secondly, note that $\delta_{cycle}$ increases with the sample size $m$ and thus reaches its maximum for $m = n$. For this value, we can interpret the formula as the expected value of the function $k \to \delta_{\mathcal{A},k}$ under a binomial distribution of parameter $1/n$.

$$\max \delta_{cycle} = \sum_{i=1}^{n} \binom{n}{k} \left(\frac{1}{n}\right)^k \left(1 - \frac{1}{n}\right)^{n-k} \delta_{\mathcal{A},k} = \mathbb{E}_{X \sim Bin(n,1/n)}(\delta_{\mathcal{A},X})$$

Thus, even if $\delta_{\mathcal{A},k}$ increases exponentially in $k$ and $\varepsilon$, we know that the moment generative function for a binomial law is $\mathbb{E}(e^{tX}) = (1 + \frac{1}{n} + \frac{1}{n}e^t)^n$ hence the term remains small.

## D.2 Histogram Computation on the Complete Graph

For discrete histogram computation on the complete graph, we propose Algorithm D.1: when receiving the token, each user perturbs his/her contribution with $L$-ary randomized response, adds it to the token and forwards the token to another user chosen uniformly at random. We have the following guarantees, which provide a privacy amplification of $O(1/\sqrt{n})$ over LDP for $T = \Omega(n)$.

**Theorem D.1.** *Let $\varepsilon \leq 1$, $\delta > 0$ and $n \geq 14^2 \log(4/\delta)$. Algorithm D.1 with $\gamma = L/(e^\varepsilon + L - 1)$ achieves an unbiased estimate of the histogram with $\gamma T$ expected random responses. Furthermore, it*

*satisfies* $(\varepsilon', (N_v + \frac{T}{n})\delta + \delta' + \hat{\delta})$*-network DP for all* $\delta', \hat{\delta} > 0$ *with*

$$\varepsilon' \leq \sqrt{\left(\frac{4T}{n} + 2\sqrt{\frac{3T}{n}\log(1/\hat{\delta})}\right)\log(1/\delta')}\frac{21\sqrt{\log(4/\delta)}}{\sqrt{n}}\varepsilon + \sqrt{\frac{2T}{n} + \sqrt{\frac{3T}{n}\log(1/\hat{\delta})}}\varepsilon(e^{21\varepsilon\sqrt{\log(4/\delta)}/\sqrt{n}}) - 1).$$

**Remark D.2.** *In the proof below, we use some approximations to obtain the simple closed-form expressions of Theorem D.1. These approximations however lead to the unnecessarily strong condition* $n \geq 14^2 \log(4/\delta)$ *and suboptimal constants in* $\varepsilon'$*. In concrete implementations, we can obtain tighter results by numerically evaluating the complete formulas.*

*Proof.* The proof follows the same steps as in the case of real summation (Appendix D.1), using the same "fictive" walk trick. We only need to adapt how we bound the privacy loss of a given cycle. More precisely, keeping the same notations as in Appendix D.1, we need to modify how we bound the modification of the privacy loss of $\mathcal{A}$. Here, $\mathcal{A}$ corresponds to the aggregation of some discrete contributions, which is equivalent to shuffling these contributions. We can therefore rely on privacy amplification by shuffling. Specifically here, we use the bound of Feldman, McMillan, and Talwar [FMT20, Theorem 3.1 therein] which is more tight and holds under less restrictive assumptions than the result of [Erl+19]. We recall the result below below.

**Theorem D.3** (Amplification by shuffling, [FMT20])**.** *For any data domain* $\mathcal{X}$*, let* $\mathcal{R}^{(i)} : \mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(i-1)} \times \mathcal{X} \to \mathcal{S}^{(i)}$ *for* $i \in [n]$ *(where* $\mathcal{S}^{(i)}$ *is the range space of* $\mathcal{R}^{(i)}$*) be a sequence of algorithms such that* $\mathcal{R}^{(i)}(z_1, \ldots, z_{i-1}, \cdot)$ *is an* $\varepsilon_0$*-DP local randomizer for all values of auxiliary inputs* $(z_1, \ldots, z_{i-1}) \in \mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(i-1)}$*. Let* $\mathcal{A}_{\mathrm{s}} : \mathcal{X}^n \to \mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(n)}$ *be the algorithm which takes as input a dataset* $(x_1, \ldots, x_n) \in \mathcal{X}^n$*, samples a uniform random permutation* $\pi$ *over* $[n]$*, then sequentially computes* $z_i = \mathcal{R}^{(i)}\left(z_1, \ldots, z_{i-1}, x_{\pi(i)}\right)$ *for* $i \in [n]$ *and outputs* $(z_1, \ldots, z_n)$*. Then for any* $\delta \in [0, 1]$ *such that* $\varepsilon_0 \leq \log\left(\frac{n}{16\log(2/\delta)}\right)$*,* $\mathcal{A}_{\mathrm{s}}$ *satisfies* $(\varepsilon_{shuff}, \delta)$*-DP with*

$$\varepsilon_{shuff} \leq \log\left(1 + \frac{e^{\varepsilon_0} - 1}{e^{\varepsilon_0} + 1}\left(\frac{8\sqrt{e^{\varepsilon_0}\log(4/\delta)}}{\sqrt{n}} + \frac{8e^{\varepsilon_0}}{n}\right)\right).$$

For clarity, we propose to use a simpler expression for $\varepsilon_{shuff}$ (Eq. D.3 below) which makes the asymptotic amplification in $O(1/\sqrt{n})$ explicit. However, it is possible to keep the initial form of Theorem D.3 for numerical applications. To derive a less tight but more tractable bound, we use the fact that $\frac{e^x - 1}{e^x + 1} \leq \frac{x}{2}$, which gives:

$$\varepsilon_{shuff} \leq \left(1 + \frac{\varepsilon_0}{2}\left(\frac{8\sqrt{e^{\varepsilon_0}\log(4/\delta)}}{\sqrt{n}} + \frac{8e^{\varepsilon_0}}{n}\right)\right).$$

We then use the hypothesis $\varepsilon_0 \leq 1$ and the concavity of the logarithm to obtain the following simple bound:

$$\varepsilon_{shuff} \leq \frac{14\sqrt{\log(4/\delta)}}{\sqrt{n}}\varepsilon_0. \tag{D.3}$$

Here, contrary to the case of real summation, amplification by shuffling is effective only for cycles whose length $m$ is large enough. To mitigate this issue, we remark that, since the $k$-ary randomized response protocol $\mathcal{A}$ satisfies $\varepsilon$-LDP, we can always bound the privacy loss of $\mathcal{A}$ by the local guarantee $\varepsilon$.

Let us assume that $m \geq 14^2 \log(4/\delta)$. This implies that $\frac{14\sqrt{\log(4/\delta)}}{\sqrt{m}}\varepsilon \leq 1$. This inequality is the hypothesis needed to simplify the expression of the privacy loss with the amplification by subsampling, as in the proof of real summation:

$$\log(1 + (1 - (1 - 1/n)^m)(e^{\frac{14\sqrt{\log(4/\delta)}}{\sqrt{m}}\varepsilon} - 1) \leq \frac{21\sqrt{\log(4/\delta)m}}{n}\varepsilon.$$

In particular, for every cycle,

$$\varepsilon_{cycle} \leq \min\Big(\frac{3m\varepsilon}{2n}, \frac{21\sqrt{\log(4/\delta)m}}{n}\varepsilon\Big),$$

where the first term corresponds to the analysis where we use amplification by subsampling and $\varepsilon$ for the privacy loss of $\mathcal{A}$, while the second one is obtained by combining amplification by subsampling with amplification by shuffling using (D.3) in the case of $m \geq 14^2 \log(4/\delta)$. We note that the second term becomes smaller when $m$ is larger than $m = 14^2 \log(4/\delta)$. In this regime, the constraint $\varepsilon \leq \log(\frac{m}{16\log(2/\delta)})$ required by Theorem D.3 is directly satisfied, as $\varepsilon \leq \log(\frac{14^2\log(4/\delta)}{16\log(2/\delta)})$ is less restrictive than $\varepsilon \leq 12.25$. As we assume that $n \geq 14^2 \log(4/\delta)$, the regime where the second term is larger exists. We see that the worst privacy loss is reached for a cycle of length $n$, for which we have:

$$\varepsilon_{cycle} \leq \frac{21\sqrt{\log(4/\delta)}}{\sqrt{n}}\varepsilon.$$

Using the above bound for the privacy loss of any cycle, we conclude by applying advanced composition as in the case of real aggregation. ■

## D.3 Proof of Theorem 7.2 (Stochastic Gradient Descent on a Complete Graph)

*Proof.* The proof tracks privacy loss using Rényi Differential Privacy (RDP) [Mir17] and leverages results on amplification by iteration [Fel+18]. We first recall the definition of RDP and the main theorems that we will use. Then, we apply these tools to our setting and conclude by translating the resulting RDP bounds into $(\varepsilon, \delta)$-DP.

Rényi Differential Privacy quantifies the privacy loss based on the Rényi divergence between the outputs of the algorithm on neighboring databases.

**Definition D.4** (Rényi divergence)**.** *Let $1 < \alpha < \infty$ and $\mu, \nu$ be measures such that for all measurable set $A$, $\mu(A) = 0$ implies $\nu(A) = 0$. The Rényi divergence of order $\alpha$ between $\mu$ and $\nu$ is defined as*

$$D_\alpha(\mu \| \nu) = \frac{1}{\alpha - 1} \log \int \left( \frac{\mu(z)}{\nu(z)} \right)^\alpha \nu(z) dz.$$

In the following, when $U$ and $V$ are sampled from $\mu$ and $\nu$ respectively, with a slight abuse of notation we will often write $D_\alpha(U||V)$ to mean $D_\alpha(\mu \| \nu)$.

**Definition D.5** (Rényi DP)**.** *For $1 < \alpha \leq \infty$ and $\varepsilon \geq 0$, a randomized algorithm $\mathcal{A}$ satisfies $(\alpha, \varepsilon)$-Rényi differential privacy, or $(\alpha, \varepsilon)$-RDP, if for all neighboring data sets $D$ and $D'$ we have*

$$D_\alpha \left( \mathcal{A}(D) \| \mathcal{A}\left(D'\right) \right) \leq \varepsilon.$$

We can introduce a notion of *Network-RDP* accordingly.

**Definition D.6** (Network Rényi DP)**.** *For $1 < \alpha \leq \infty$ and $\varepsilon \geq 0$, a randomized algorithm $\mathcal{A}$ satisfies $(\alpha, \varepsilon)$-network Rényi differential privacy, or $(\alpha, \varepsilon)$-NRDP, if for all pairs of distinct users $u, v \in V$ and all pairs of neighboring datasets $D \sim_u D'$, we have*

$$D_\alpha \left( \mathcal{O}_v(\mathcal{A}(D)) \| \mathcal{O}_v \left( \mathcal{A}(D') \right) \right) \leq \varepsilon.$$

As in classic DP, there exists composition theorems for RDP, see [Mir17]. We will use the following.

**Proposition D.7** (Composition of RDP)**.** *If $\mathcal{A}_1, \ldots, \mathcal{A}_k$ are randomized algorithms satisfying $(\alpha, \varepsilon_1)$-RDP, $\ldots$, $(\alpha, \varepsilon_k)$-RDP respectively, then their composition $(\mathcal{A}_1(S), \ldots, \mathcal{A}_k(S))$ satisfies $(\alpha, \sum_{l=1}^k \varepsilon_l)$-RDP. Each algorithm can be chosen adaptively, i.e., based on the outputs of algorithms that come before it.*

Finally, we can translate the result of the RDP by using the following result.

**Proposition D.8** (Conversion from RDP to DP [Mir17]). *If $\mathcal{A}$ satisfies $(\alpha, \varepsilon)$-Rényi differential privacy, then for all $\delta \in (0,1)$ it also satisfies $\left(\varepsilon + \frac{\log(1/\delta)}{\alpha - 1}, \delta\right)$ differential privacy.*

Privacy amplification by iteration [Fel+18] captures the fact that for algorithms that consist of *iterative contractive updates*, not releasing the intermediate results improve the privacy guarantees for the final result. An important application of this framework is Projected Noisy Stochastic Gradient Descent (PNSGD) in the centralized setting, where the trusted curator only reveals the final model. More precisely, when iteratively updating a model with PNSGD, any given step is hidden by subsequent steps (the more subsequent steps, the better the privacy). The following result from [Fel+18] (Theorem 23 therein) formalizes this.

**Theorem D.9** (Rényi differential privacy of PNSGD). *Let $\mathcal{W} \in \mathbb{R}^d$ be a convex set, $\mathcal{X}$ be an abstract data domain and $\{f'; x)\}_{x \in \mathcal{X}}$ be a family of convex $L$-Lipschitz and $\beta$-smooth function over $\mathcal{K}$. Let PNSGD$(D, w_0, \eta, \sigma)$ be the algorithm that returns $w_n \in \mathcal{W}$ computed recursively from $w_0 \in \mathcal{W}$ using dataset $D = \{x_1, \ldots, x_n\}$ as:*

$$w_{t+1} = \Pi_{\mathcal{W}}(w_t - \eta(\nabla f(w_t; x_{t+1}) + Z)), \quad \text{where } Z \sim \mathcal{N}(0, \sigma^2 I_d).$$

*Then for any $\eta \leq 2/\beta, \sigma > 0, \alpha > 1, t \in [n]$, starting point $w_0 \in \mathcal{K}$ and $D \in \mathcal{X}^n$, PNSGD satisfies $(\alpha, \frac{\alpha\varepsilon}{n+1-t})$-RDP for its $t$-th input, where $\varepsilon = \frac{2L^2}{\sigma^2}$ .*

In our context, we aim to leverage this result to capture the privacy amplification provided by the fact that a given user $v$ will only observe information about the update of another user $u$ after some steps of the random walk. To account for the fact that this number of steps will itself be random, we will use the so-called weak convexity property of the Rényi divergence [Fel+18].

**Proposition D.10** (Weak convexity of Rényi divergence). *Let $\mu_1, \ldots, \mu_m$ and $\nu_1, \ldots, \nu_m$ be probability distributions over some domain $\mathcal{Z}$ such that for all $i \in [m]$, $D_\alpha(\mu_i \| \nu_i) \leq c/(\alpha - 1)$ for some $c \in (0, 1]$. Let $\rho$ be a probability distribution over $[m]$ and denote by $\mu_\rho$ (resp. $\nu_\rho$) the probability distribution over $\mathcal{Z}$ obtained by sampling $i$ from $\rho$ and then outputting a random sample from $\mu_i$ (resp. $\nu_i$). Then we have:*

$$D_\alpha(\mu_\rho \| \nu_\rho) \leq (1 + c) \cdot \mathop{\mathbb{E}}_{i \sim \rho}\left[D_\alpha(\mu_i \| \nu_i)\right].$$

We now have all the technical tools needed to prove our result. Let us denote by $\sigma^2 = \frac{8L^2 \log(1.25/\delta)}{\varepsilon^2}$ the variance of the Gaussian noise added at each gradient step in Algorithm 7.2. Let us fix two distinct users $u$ and $v$. We aim to quantify how much information about the private data of user $u$ is leaked to $v$ from the visits of the token. In contrast to the proofs of Theorem 7.1 (real summation) and Theorem D.1 (discrete histogram computation), we will

reason here on the privacy loss induced by each contribution of user $u$, rather than by each visit of the token through $v$.

Let us fix a contribution of user $u$ at some time $t_1$. The view $\mathcal{O}_v$ of user $v$ on the entire procedure is defined as in the proof of Theorem 7.1. Note that the token values observed before $t_1$ do not depend on the contribution of $u$ at time $t_1$. Let $t_2 > t_1$ be the first time that $v$ receives the token posterior to $t_1$. It is sufficient to bound the privacy loss induced by the observation of the token at $t_2$: indeed, by the post-processing property of DP, no additional privacy loss with respect to $v$ will occur for observations posterior to $t_2$.

By definition of the random walk, $t_2$ follows a geometric law of parameter $1/n$, where $n$ is the number of users. Additionally, if there is no time $t_2$ (which can be seen as $t_2 > T$), then no privacy loss occurs. Let $Y_v$ and $Y_v$ be the distribution followed by the token when observed by $v$ at time $t_2$ for two neighboring datasets $D \sim_u D'$ which only differ in the dataset of user $u$. For any $t$, let also $X_t$ and $X'_t$ be the distribution followed by the token at time $t$ for two neighboring datasets $D \sim_u D'$. Then, we can apply Proposition D.10 to $D_\alpha(Y_v||Y'_v)$ with $c = 1$, which is ensured when $\sigma \geq L\sqrt{2\alpha(\alpha-1)}$, and we have:

$$D_\alpha(Y_v||Y'_v) \leq (1+1)\mathbb{E}_{t\sim\mathcal{G}(1/n)} D_\alpha(X_t||X'_t).$$

We can now bound $D_\alpha(X_t||X'_t)$ for each $t$ using Theorem D.9 and obtain:

$$\begin{aligned} D_\alpha(Y_v||Y'_v) &\leq \textstyle\sum_{t=1}^{T-t_1} \frac{1}{n}(1-\frac{1}{n})^t \frac{2\alpha L^2}{\sigma^2 t} \\ &\leq \textstyle\frac{2\alpha L^2}{\sigma^2 n} \sum_{t=1}^{\infty} \frac{(1-1/n)^t}{t} \\ &\leq \textstyle\frac{2\alpha L^2 \log n}{\sigma^2 n}. \end{aligned}$$

To bound the privacy loss over all the $T_u$ contributions of user $u$, we use the composition property of RDP, leading to the following Network RDP guarantee.

**Proposition D.11.** *Let $\alpha > 1$, $\sigma \geq L\sqrt{2\alpha(\alpha-1)}$ and $T_u$ be maximum number of contributions of a user. Then Algorithm 7.1 satisfies $(\alpha, \frac{4T_u\alpha L^2 \log n}{\sigma^2 n})$-Network Rényi DP.*

We can now convert this result into an $(\varepsilon_c, \delta_c)$-DP statement using Proposition D.8. This proposition calls for minimizing the function $\alpha \to \varepsilon_c(\alpha)$ for $\alpha \in (1, \infty)$. However, recall that from our use of the weak convexity property we have the additional constraint on $\alpha$ requiring that $\sigma \geq L\sqrt{2\alpha(\alpha-1)}$. This creates two regimes: for small $\varepsilon_c$ (i.e, large $\sigma$ and small $T_u$), the minimum is not reachable, so we take the best possible $\alpha$ within the interval, whereas we have an optimal regime for larger $\varepsilon_c$. This minimization can be done numerically, but for simplicity of exposition we can derive a suboptimal closed form which is the one given in Theorem 7.2.

To obtain this closed form, we reuse the result of [Fel+18] (Theorem 32 therein). In particular, for $q = \max\left(\frac{2T_u \log n}{n}, 2\log(1/\delta_c)\right)$, $\alpha = \frac{\sigma\sqrt{\log(1/\delta_c)}}{L\sqrt{q}}$ and $\varepsilon_c = \frac{4L\sqrt{q\log(1/\delta_c)}}{\sigma}$, the conditions $\sigma \geq L\sqrt{2\alpha(\alpha-1)}$ and $\alpha > 2$ are satisfied. Thus, we have a bound on the privacy loss which holds the two regimes thanks to the definition of $q$.

Finally, we bound $T_u$ by $N_u = \frac{T}{n} + \sqrt{\frac{3T}{n}\log(1/\hat{\delta})}$ with probability $1-\hat{\delta}$ as done in the previous proofs for real summation and discrete histograms. Setting $\varepsilon' = \varepsilon_c$ and $\delta' = \delta_c + \hat{\delta}$ concludes the proof. ■

**Remark D.12** (Tighter numerical bounds)**.** *As mentioned in the proof, we can compute a tighter bound for small $\sigma$ when the optimal $\alpha$ violates the constraints on $\sigma$. In this case, we set $\alpha$ to its limit such that $\sigma = L\sqrt{2\alpha(\alpha-1)}$ and deduce a translation into $(\varepsilon_c, \delta_c)$-differential privacy. This is useful when $q \neq \frac{2N_u \log n}{n}$, i.e., situations where the number of contributions of a user is smaller than the number of users.*

*In particular, we use this method in our experiments of Section 7.2.4. In that case, we have a fixed $(\varepsilon_c, \delta_c)$-DP constraint and want to find the minimum possible $\sigma$ that ensures this privacy guarantee. We start with a small candidate for $\sigma$ and compute the associated privacy loss as explained above. We then increase it iteratively until the resulting $\varepsilon_c$ is small enough.*

## D.4 Lifting the Assumption of Hidden Sender/Receiver

In the analysis of Section 7.2, we assumed that a user does not know the identity of the previous (sender) or next (receiver) user in the walk. We discuss here how we can lift this assumption. Our approach is based on separately bounding the privacy loss of contributions that are adjacent to a given user (*spotted contributions*), as these contributions do not benefit from any privacy amplification if the identity of the sender/receiver is known. We first compute the privacy loss resulting from spotted contributions, then discuss in which regimes this term becomes negligible in the total theoretical privacy loss, and finally how to deal with it empirically.

**Definition D.13** (Spotted contribution)**.** *For a walk on the complete graph, we define a spotted contribution of $u$ with respect to $v$ as a contribution of $u$ that is directly preceded or followed by a contribution of $v$.*

A spotted contribution has a privacy loss bounded by $\varepsilon$, as we still have the privacy guarantee given by the local randomizer, but no further amplification of privacy. Thus, we need to bound the number of contributions for a given vertex $u$ to be spotted by another user $v$. As in the proofs of Theorem 7.1, Theorem 7.2 and Theorem D.1, we bound the number of contributions of $u$ by $N_u$ using Chernoff. Now, for each of these contributions, the probability of being spotted

is $2/n$, so the number of spotted contributions follows a binomial law of parameter $\mathcal{B}(N_u, 2/n)$. We then use once again Chernoff to bound the number of spotted contributions with probability $\tilde{\delta}$, and use either simple or advanced composition. This leads to the following bound for the privacy loss associated with spotted contributions.

**Proposition D.14** (Privacy loss of spotted contributions)**.** *For a random walk with $N_u$ contributions of user $u$, the privacy loss induced by spotted contributions is bounded with probability $1 - \tilde{\delta}$ by:*

- $\varepsilon_s = \sqrt{\left(\frac{2N_u}{n} + \sqrt{\frac{6N_u}{n}\log(1/\tilde{\delta})}\right)\log(1/\delta')}\varepsilon + \left(\frac{2N_u}{n} + \sqrt{\frac{6N_u}{n}\log(1/\tilde{\delta})}\right)\varepsilon(e^{\varepsilon} - 1)$ *with advanced composition,*
- $\varepsilon_s = \left(\frac{2N_u}{n} + \sqrt{\frac{6N_u}{n}\log(1/\tilde{\delta})}\right)\varepsilon$ *with simple composition.*

The above term (along with $\tilde{\delta}$) should be added to the total privacy loss to take into account the knowledge of the previous and next user. The difficulty comes from the fact that the number of spotted contributions has a high variance if the number of contributions per user is small compared to the number of users. We already observed this when bounding the number of contributions per user, where the worst case is far from the expected value (see Figure 7.1(a)). Here, the price to pay is higher as the square root dominates the expression in the regime where $T = o(n^2)$. However, for $T = \Omega(n^2)$, the spotted contribution term becomes negligible and we recover the same order of privacy amplification as in Theorem 7.1, Theorem 7.2 and Theorem D.1.

The derivations above provide a way to bound the impact of spotted contributions theoretically, but we can also deal with it empirically. In practical implementations, one can also enforce a bound on the number of times that an edge can be used, and dismiss it afterwards, with limited impact on the total privacy loss. Another option to keep the same formal guarantees is to replace real contributions with only noise when an edge has exceeded the bound. These "fake contributions" seldom happen in practice and thus do not harm the convergence.

## D.5 Additional Experiments

We run experiments to investigate the empirical behavior of our approach for the task of discrete histogram computation on the complete graph by leveraging results on privacy amplification by shuffling. Here, we have used the numerical approach from [Bal+19b] to tightly measure the effect of amplification by shuffling based on the code provided by the authors.[2] Figure D.1 con-

[2] https://github.com/BorjaBalle/amplification-by-shuffling

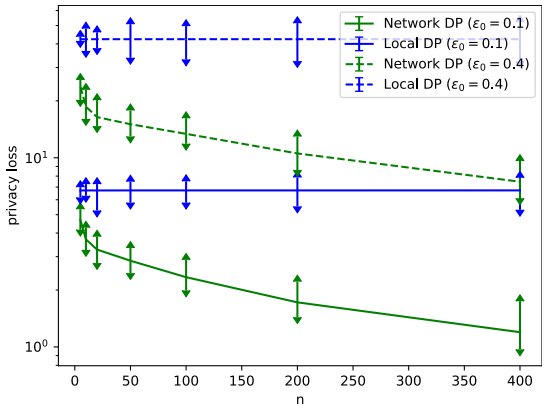



**Figure D.1** – Comparing network and local DP on the task of computing discrete histograms. The results are obtained for $T = 100n$ (i.e., the expected number of contributions per user is $100$). The value of $\varepsilon_0$ rules the amount of local noise added to each contribution (i.e., each single contribution taken in isolation satisfied $\varepsilon_0$-LDP). Curves report the average privacy loss across all pairs of users and all $10$ random runs, while their error bars give the best and worst cases.

firm that the empirical gains from privacy amplification by decentralization are also significant for this task.

## D.6 Proof of Theorem 7.6

We adapt the proof of Theorem 5 of [Eve23]. Because constants matters in differential privacy, we detail the steps needed to explicit these constants. We recall here the keys steps and definition and how we adapt the proofs to allow the addition of Gaussian noise. We keep the same definitions, except that the sequence of token iterates is now defined by

$$x_{t+1} = x_t - \gamma(g_t + \eta_t)\,.$$

This does not impact Lemma 8, which only relies on the graph properties and the function at the optimum. Up to the transposition of notation, we have for any $T \geqslant 1$:

$$\mathbb{E}\left[\left\|\sum_{t<T} \nabla f_{v_t}\left(x^\star\right)\right\|^2\right] \leqslant T\zeta_\star^2 + \zeta_\star^2 \sum_{t<T} \mathrm{d_{TV}}\left(P^t_{v_0,}, \pi^\star\right) + 2\zeta_\star^2 \sum_{s<t<T} \mathrm{d_{TV}}(t-s), \tag{D.4}$$

where $\mathrm{d_{TV}}(r) = \sup\left\{\mathrm{d_{TV}}\left(\left(P^r\right)_{v,}, \pi^\star\right), v \in \mathcal{V}\right\}$ for $r \in \mathbb{N}$, so that:

$$\mathbb{E}\left[\left\|\sum_{t<T} \nabla f_{v_t}\left(x^\star\right)\right\|^2\right] \leqslant \zeta_\star^2\left(4\tau_{\mathrm{mix}}\left(1/4\right) + T\left(1 + 8\tau_{\mathrm{mix}}\left(1/4\right)\right)\right). \tag{D.5}$$

Next, we can transform Lemma 9 into

$$\mathbb{E}_\eta(\|x_{t+1} - y_{t+1}\|^2) \leq (1-\gamma\mu)\mathbb{E}_\eta(\|x_t - y_t\|^2) + \gamma L\left\|y_t - x^*\right\|^2 + \gamma^2(d\sigma^2 + \sigma_{sgd}^2)\,.$$

where the sequence of $y_t$ and $y_{t+1}$ satisfies the relation.

$$y_{t+1} = y_t - \gamma\nabla f_{v_t}(x^*)$$

By applying the formula recursively, we obtain:

$$\mathbb{E}_\eta\left\|x_T - y_T\right\|^2 \leqslant (1-\gamma\mu)^T\left\|x_0 - y_0\right\|^2 + \sum_{t<T}(1-\gamma\mu)^{T-t}\left(\gamma L\left\|y_t - x^\star\right\|^2 + \gamma^2(d\sigma^2 + \sigma_{sgd}^2)\right),$$

where we use $\mathbb{E}_\eta$ to denote the expected value with respect to the privacy noise.

By instantiating the $y$ sequence as done in the non-private version with the $x$, we recover nearly the same formula, with $y_t = x^*$. The first term can be handled as in the non-private case,

while the second term has an additional sum:

$$\mathbb{E}_\eta \left\|x_T - x^\star\right\|^2 \leqslant 2(1-\gamma\mu)^T \left( \mathbb{E}\left\|x_0 - x^\star\right\|^2 + \gamma^2 \mathbb{E}\left[ \left\| \sum_{s<T} \nabla f_{v_s}(x^\star) \right\|^2 \right] \right) + \sum_{t<T} (1-\gamma\mu)^{T-t} \left( \gamma^3 L \mathbb{E}\left[ \left\| \sum_{t\leqslant s<T} \nabla f_{v_s}(x^\star) \right\|^2 \right] + \gamma^2(d\sigma^2 + \sigma_{sgd}^2) \right).$$

As $\sum_{t<T}(1-\gamma\mu)^{T-t} < \frac{1}{\gamma\mu}$, and the other terms remain identical, we have:

$$\mathbb{E}_\eta \left\|x_T - y_T\right\|^2 \leqslant 2(1-\gamma\mu)^T \left\|x_0 - x^\star\right\|^2 + \frac{3\gamma L}{\mu^2} C \tau_{\text{mix}} \zeta_\star^2 + \frac{\gamma(d\sigma^2 + \sigma_{sgd}^2)}{\mu}.$$

with $C = 13$.

We conclude by plugging back the following $\gamma$ (as in [Eve23]) in the previous formula:

$$\gamma = \min\left( \frac{1}{L}, \frac{1}{T\mu} \log\left( T \frac{\left\|x_0 - x^\star\right\|^2}{\frac{39}{\mu^2} C \tau_{\text{mix}} \zeta_\star^2} \right) \right).$$

**Remark D.15.** *As long as $\sigma^2 \leq \frac{39 L \tau_{\text{mix}} \zeta_*^2}{d\mu}$, the noise due to privacy is smaller than the one due to the randomness of the walk.*

## D.7 Privacy Proofs

Let $u, v$ be two distinct nodes. To prove Theorem 7.7, we see the privacy loss $\varepsilon_{u\to v}$ as the composition of the privacy loss induced by each of the contributions of node $u$. Thus, we first bound the privacy loss for one contribution $\varepsilon_{u\to v}^{\text{single}}$ . Let us denote by $t_c$ the time step where this contribution is made (i.e., the token is at node $u$ at time $t_c$). For $t \leq t_c$, there is no privacy leakage. Let us denote by $t_l$ the first $t \geq t_c$ where the token is held by $v$. By the post-processing property of differential privacy, the steps after $t_l$ will not yield additional leakage for the contribution of time $t_c$. Hence, we only need to bound the privacy leakage at time $t_l$. This leakage depends on the number of steps between $t_c$ and $t_l$. We use the weak convexity property of the Rényi divergence to decompose our privacy loss.

**Lemma D.16** (Weak convexity of Rényi divergence)**.** *Let $\mu_1, \ldots, \mu_n$ and $\nu_1, \ldots, \nu_n$ be probability distributions over some domain $\mathcal{Z}$ such that for all $i \in [n], \mathrm{D}_\alpha(\mu_i \| \nu_i) \leq c/(\alpha - 1)$ for some $c \in (0, 1]$. Let $\rho$ be a probability distribution over $[n]$ and denote by $\mu_\rho$ (respectively, $\nu_\rho$) the probability distribution over $\mathcal{Z}$ obtained by sampling $i$ from $\rho$ and then outputting a random sample*

*from $\mu_i$ (respectively, $\nu_i$). Then*

$$\mathrm{D}_\alpha\left(\mu_\rho \| \nu_\rho\right) \leq (1+c) \cdot \mathop{\mathbb{E}}_{i\sim\rho}\left[\mathrm{D}_\alpha\left(\mu_i \| \nu_i\right)\right] . \tag{D.6}$$

We can thus partition according to the length of the walk and write the privacy guarantee depending on $T$ the total number of steps and $\beta(i)$ a function bounding the privacy loss occurring for seeing the token $i$ steps after the node contribution:

$$\varepsilon_{u\to v}^{\mathsf{single}} \leq (1+c) \sum_{i=1}^{T} \mathbb{P}(u \to v \text{ after } i \text{ steps}) \beta(i) . \tag{D.7}$$

The probability of the path of $t$ steps between $u$ and $v$ can be easily extracted from the power of the transition matrix $W$. We thus obtain the generic formula.

**Lemma D.17.** *Let $\beta(i)$ be a bound on the privacy loss occurring for seeing the token $i$ steps, and $T$ the total number of steps. Then, the following holds:*

$$\varepsilon_{u\to v}^{\mathsf{single}} \leq \sum_{i=1}^{T} W_{uv}^{i} 2\beta(i) . \tag{D.8}$$

We recognize in this formula the communicability (Definition 7.10) with $c_i = 2\beta(i)$.

We can now compute the function $\beta$ by resorting to privacy amplification by iteration (Theorem 3.11). As we apply the Gaussian mechanism at each step, we have $s_1 = \frac{\alpha}{2\sigma^2}$ and $s_j = 0$ for $0 < j \leq i$. We thus take all $a_j = \frac{\alpha}{2\sigma^2 i}$ in Theorem 3.11. This gives the bound $\beta(i) = \frac{\alpha}{2\sigma^2 i}$, which corresponds to setting $c_i = \alpha\sigma^2 i$.

We now focus on how to simplify and compute this formula when the transition matrix is bistochastic and symmetric. Since the matrix is symmetric by assumption, we can apply the spectral theorem to write:

$$W = \sum_{i=1}^{n} \lambda_i \phi_i \phi_i^\top , \tag{D.9}$$

Furthermore, since the matrix is bistochastic, $\lambda_1 = 1 > \lambda_2 \geq \cdots \leq \lambda_n > -1$ and the eigenvector associated to the first eigenvalue is $\frac{1}{\sqrt{n}}\mathbb{1}$. Hence, we can isolate the first term and plug into (D.8) to get:

$$\varepsilon_{u\to v}^{\mathsf{single}} \leq \sum_{t=1}^{T} \frac{\alpha}{\sigma^2 t} \frac{1}{n} + \sum_{i=2}^{n} \sum_{t=1}^{T} \frac{\alpha}{\sigma^2 t} \lambda_i^t \phi_i(u) \phi_i(v) . \tag{D.10}$$

In these sums, we isolate $\sum_{t=1}^{T} \lambda_i^t / t$. Noticing that the sum converges for all these eigenvalues, we can rewrite it as $\sum_{t=1}^{T} \lambda_i^t / t = -\log(1-\lambda_i) + \mathcal{O}(\lambda_i^T)$. We use the integral test for convergence to replace the sum by the logarithm.

This gives us the privacy loss for a single contribution. In order to conclude, we compose the privacy loss of each of the $N_u$ contributions of node $u$, leading to, for any $\sigma^2 \geq 2\alpha(\alpha-1)$ :

$$\varepsilon_{u\to v} \leq \frac{\alpha N_u \log(T)}{\sigma^2 n} - \frac{\alpha N_u}{\sigma^2}\log(I - W + \frac{1}{n}\mathbb{1}\mathbb{1}^\top)_{uv} + \mathcal{O}(\lambda_2^T)\,. \tag{D.11}$$

As $|\lambda_2| < 1$, its power decreases exponentially fast with the number of steps and thus is negligible with respect to the other terms.

The average number of contributions is $T/n$ as the transition matrix is bi-stochastic. We can then upper bound the real number of contributions with high probability, for example by using Theorem 12.21 of [LP17]. The small probability of exceeding the upper bound can be added to $\delta$ when converting from RDP to $(\varepsilon, \delta)$-differential privacy.

**Remark D.18.** *The upper bound on $N_u$ tends to be large for "cryptographically" small $\delta$. An efficient way to avoid this issue in practical implementations is to force a tighter bound on the maximum number of contributions by each node. After a node reaches its maximum number of contributions, if the token passes by the node again, the node only adds noise. We use this trick in numerical experiments.*

### D.7.1 Adaptation to the case without sender anonymity

For bounding the privacy loss of a single contribution, we consider above the value of the privacy loss occurring at time $t_l$. However, this is computed without taking into account the knowledge of where the token comes from, for example, if $v$ can know which of its neighbors sent him the token. This computation is thus justified only in the specific case where the anonymity of the sender is ensured, e.g., by resorting to mix networks [SP06] or anonymous routing [DMS04].

If this is not the case, then we should *a priori* use the conditional probability towards the position of the token at time $t_l - 1$. However, there is no close form for this in general for a graph. To fix this issue, we consider that the last step for reaching $v$ is only a post-processing of the walk reaching one of its neighbors.

For each of the neighbors, the previous formula applies. We obtain different values for the various neighbors, so a worst-case analysis consists in taking the max over this set. Denoting by $\widetilde{\varepsilon_{uv}^{\mathsf{single}}}$ the privacy loss when a node $v$ knows the neighbors from which it received the token, we have

$$\widetilde{\varepsilon_{uv}^{\mathsf{single}}} \leqslant \max_{w\in\mathcal{N}_v} \varepsilon_{uw}^{\mathsf{single}}$$

This approach allows to keep a closed form for the matrix and just add a max step. However, this analysis is not tight and may lead to a significant cost in some scenarios. A simple example

is the special case where the nodes $u$ and $v$ are neighbors. It effectively assumes that the transition between $u$ and $v$ is always direct, which may not always be the case.

We provide in Figure D.2 the equivalent of Figure 7.3(a) by taking the max below. As expected, amplification is smaller for close nodes, but the curves have the same asymptotic.

## D.8 Additional Numerical Experiments

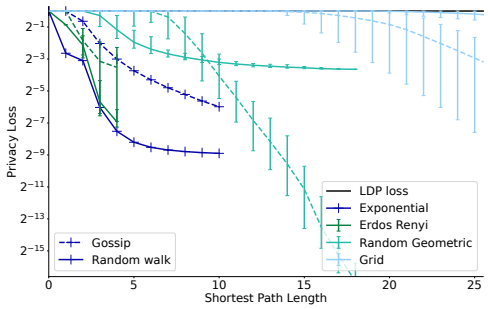



**Figure D.2** – Comparison of privacy loss for random walks when nodes know who send them the token in bold lines and gossip in dashed lines for the same synthetic graphs with $n = 2048$. Privacy amplification is visible even for close neighbors but the decay is slower than for gossip.

In this section, we include other examples of graphs that illustrate how the privacy loss matches the graph structure.

A classic synthetic graph to exhibit subgroup is the stochastic block model, where each edge follows an independent Bernoulli random variable. The parameter of the law depends from a matrix encoding the relation between the clusters. As for other graphs, the privacy loss is similar to communicability and shows different level of privacy within and outside a group Figure D.3(b), so that a node sees the major part of its privacy loss occurring within its group (Figure D.3(a).

We report other examples of privacy loss for a node chosen at random in Facebook Ego graphs. Once again, these results illustrate that our privacy loss guarantees match the existing clusters of the graph (Figure D.4).

Intuitively, compare to gossip algorithms where the updates only slowly flows in the graph, the random walk should mix quite easily heterogeneous data. while we do not derive mathematical guarantees on heterogeneity, we illustrate this idea with the following numerical experiment. We generate a synthetic geometric random graph and compare two scenarii (Figure D.5). On the first trial, the data is position-dependent, which generate heterogeneity as close nodes also have close data point. We then shuffle randomly the data to destroy the heterogeneity and compare the convergence of the two in Figure D.6.

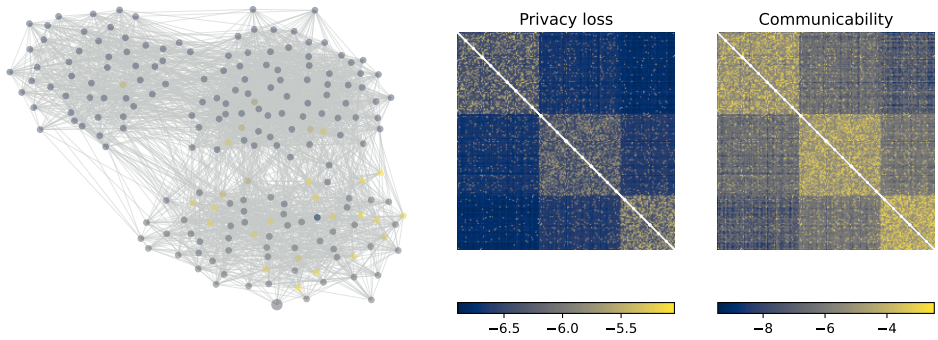



**Figure D.3** – Stochastic Block Model with 200 nodes in three clusters $(75, 75, 50)$ and probability matrix $[[0.25, 0.05, 0.02], [0.05, 0.35, 0.07], [0.02, 0.07, 0.40]]$. The privacy loss matrix recovers the different blocks.

## D.9 Collusion

The results of our work assume that nodes are separated entities that do not share information outside the protocols. One could however claim that a fraction of nodes can collude and share information between them. In this case, if we denote $F \subset V$ the fraction of the colluded nodes, for a given contribution done by $u$ what matters is the first time that one of the node of $F$ is reached afterwards. More precisely, we can derive the privacy loss as

$$\varepsilon_{u\to F}^{single} \leq \sum_{i=1}^{T} \left( \sum_{v\in F} W_{uv}^{i} \right) \alpha\sigma^2 i \,. \tag{D.12}$$

where the term between parenthesis corresponds to the probability to reach $F$ in exactly $i$ steps from $u$. By reorganizing these terms, we obtain the upper bound:

$$\varepsilon_{u\to F} \leq \sum_{v\in F} \varepsilon_{u\to v} \tag{D.13}$$

This term corresponds also to the formula one would obtain from basic composition. Hence, our analysis does not allow to avoid the degradation of the privacy guarantees to collusion. Note that if all the colluded nodes are far away from $u$, it is still possible to derive non trivial guarantees compare to what would give the bound of local differential privacy. In comparison to gossip where the privacy loss decrease is sharper with distance, the cases where the amplification remains are scarcer. This is a fundamental limitation of amplification by decentralization, that was already pointed out in [CB22; Cyf+22].

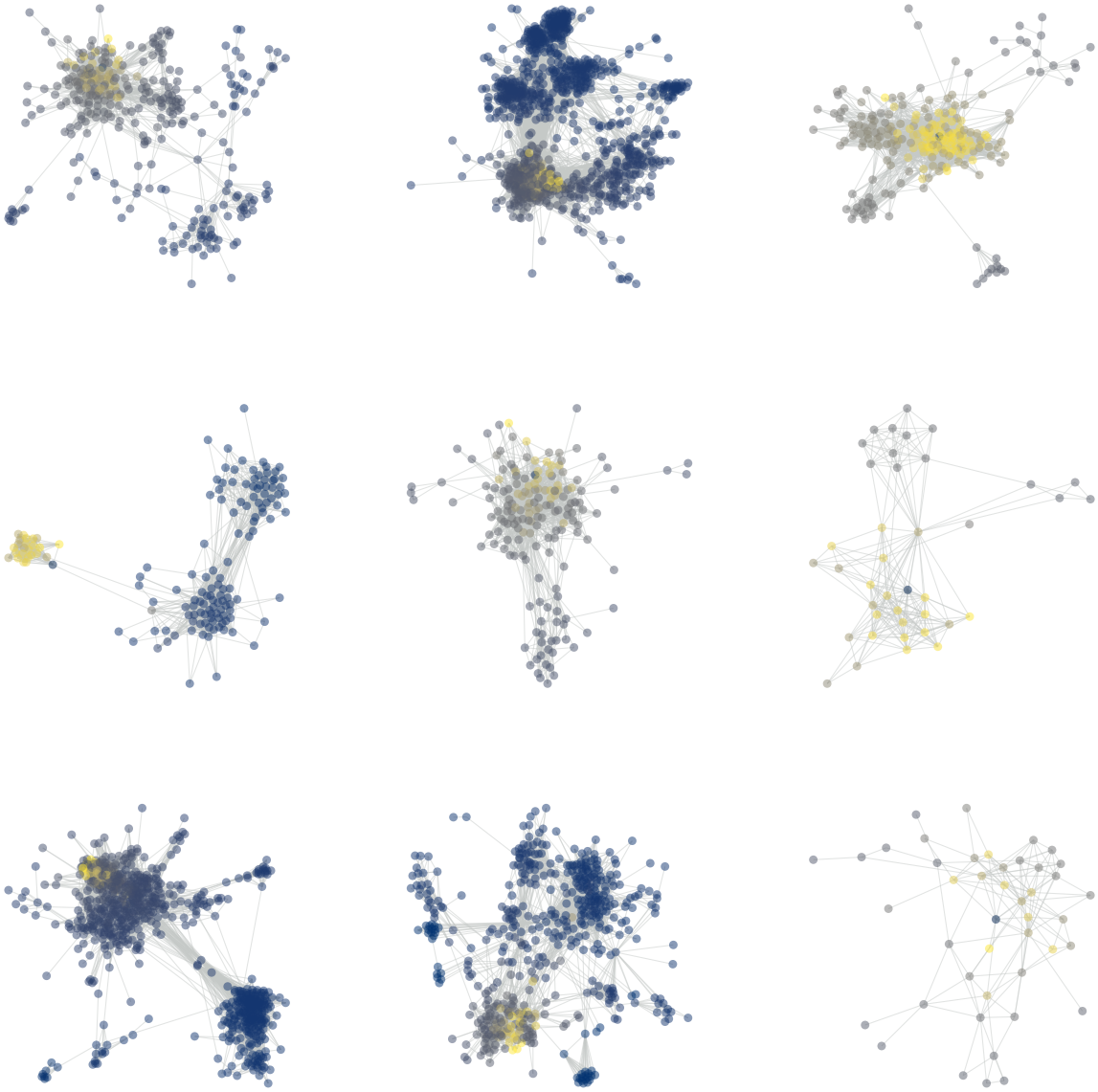


**Figure D.4** – Privacy loss on the 9 other Facebook Ego graphs, following the same methodology as in Figure 7.5(a).

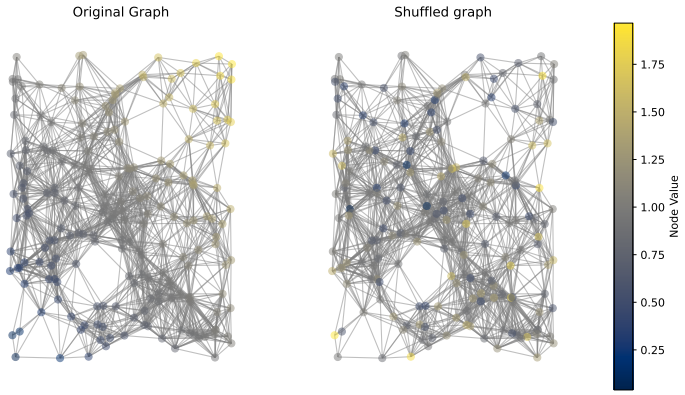
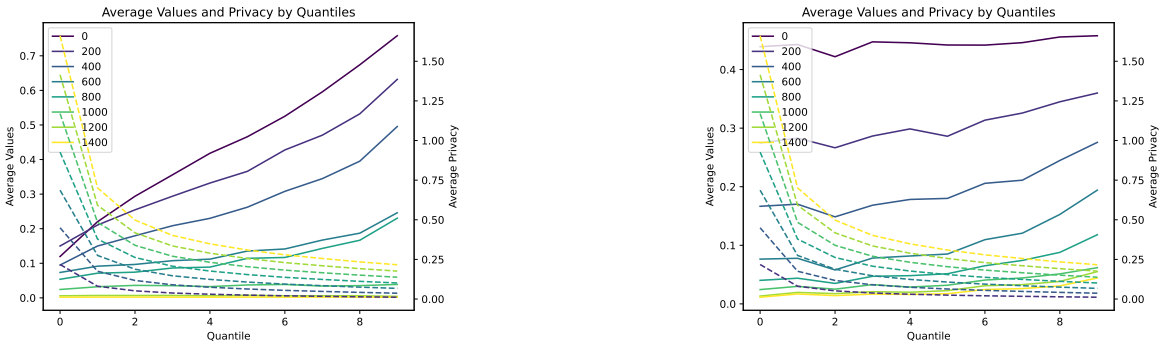



**Figure D.5** – Geometric random graph with 200 nodes. On the left, the label is given by the sum of the coordinates, providing heterogeneity in the graph. On the right the same graph has its label shuffled




(a) With original labels

(b) With shuffled labels

**Figure D.6** – We compute the quantiles of the euclidean distance between node. For each quantile, we report the mean on all the pair of nodes of the quantiles for the privacy loss and for the distance between the current estimates. We report different time step across the learning. The privacy loss is identical in both cases. At the beginning of the learning, the homogeneous case has smaller average values heterogeneity, but the difference reduces with the learning

## D.10 Refined Privacy Bounds for Specific Graphs

### D.10.1 Useful Auxiliary Results

We first collect some auxiliary results that we use in our bounds.

**Proposition D.19.** *For any $x \in (0,1)$, we have*

$$\sum_{p \text{ is odd}} \frac{x^p}{p} = \frac{1}{2}\log\frac{1+x}{1-x} \quad \textit{and} \quad \sum_{p \text{ is even}} \frac{x^p}{p} = -\log(1-x^2)\,.$$

*Proof.* Recall that

$$\log(1+x) = x + \frac{x^2}{2} + \frac{x^3}{3} + \cdots, \quad \text{and} \quad \log(1-x) = -x + \frac{x^2}{2} - \frac{x^3}{3} + \cdots$$

Now $\log(1+x) - \log(1-x)$ gives the first bound and $\log(1-x) + \log(1+x)$ gives the second bound. This completes the proof. ■

**Proposition D.20** (Godsil and Royle [GR01])**.** *Let $1 \le d \le n-1$ be an integer. Then for any $d$-regular graph $G$, the eigenvectors of the Laplacian and those of the adjacency matrix of $G$ coincide.*

### D.10.2 Privacy Loss for Specific Graphs

**Complete graph.** The transition matrix is exactly $\frac{1}{n}\mathbb{1}\mathbb{1}^\top$ and thus we recover exactly the same formula as in [CB22]. In particular, there is only one non-zero eigenvalue with magnitude 1 with an all-one vector as the corresponding eigenvector. In particular,

$$\varepsilon_{u\to v} \le \frac{\alpha}{\sigma^2}\sum_{i=1}^{T} W^i_{uv}\frac{1}{i} = \frac{\alpha}{\sigma^2}\sum_{i=1}^{T}\frac{1}{i}\left(\sum_{j=1}^{n}\lambda_j^i v_j v_j^\top\right)_{uv} = \frac{\alpha}{n\sigma^2}\sum_{i=1}^{T}\frac{1}{i} \le \frac{\alpha\log(T)}{n\sigma^2}$$

**Ring graph.** To ensure aperiodic Markov chain, the transition matrix should be in the form $aI + b(J + J^\top), a + 2b = 1$.

The adjacency matrix $A_R$ of the ring graph $R$ is a circulant matrix. Therefore, all its eigenvectors are just the Fourier modes [HJ12]:

$$\phi(\omega_k) = \begin{pmatrix} 1 \\ \omega_k \\ \omega_k^2 \\ \vdots \\ \omega_k^{n-1} \end{pmatrix},$$

where $\omega_k^n = 1$ is the $n$-th root of unity, i.e, $e^{2\pi\imath k/n}$ for $1 \le k \le n$. This can be seen by noting that multiplication with a circulant matrix gives a convolution. In the Fourier space, convolutions become multiplication. Hence the product of a circulant matrix with a Fourier mode yields a multiple of that Fourier mode, which by definition is an eigenvector.

The eigenvalues can be then computed as

$$\omega_k + \omega_k^{-1} = 2\cos(2\pi k/n) \quad \text{for} \quad 0 \le k \le n-1\,.$$

Computing $\phi(\omega)\phi(\omega)^\top$ is straightforward. The $(u,v)$-th entry would be just $\omega^{(u+v-2) \mod n}$. Recall that

$$\varepsilon_{u\to v} \le \frac{\alpha}{\sigma^2}\sum_{i=1}^{T}\frac{W^i_{uv}}{i}\,.$$

For ease of calculation, let us assume that $a = b = \frac{1}{3}$. Then $W = \frac{A_R}{3} + \frac{\mathbb{I}}{3}$, where $A_R$ is a binary matrix with $(i,j)$-th entry 1 only when $|i-j| = 1$. Then the eigenvalues of $W$ are given by $\frac{2\cos(2\pi k/n)+1}{3}$ for $0 \le k \le n-1$. Let $a = (u+v-2) \mod n$. Hence,

$$\begin{aligned}
\varepsilon_{u\to v} &\le \frac{\alpha}{n\sigma^2}\sum_{t=1}^{T}\left(\frac{e^{2\pi\imath a/n}}{t} + \sum_{k=1}^{n-1}\frac{1}{t}\left(\frac{2\cos(2\pi k/n)+1}{3}\right)^t e^{2\pi\imath ak/n}\right)\\
&\le \frac{\alpha}{n\sigma^2}\sum_{t=1}^{T}\left(\frac{e^{2\pi\imath a/n}}{t} + \sum_{k=1}^{n-1}\frac{1}{t}\left(\frac{4\cos^2(\pi k/n)-1}{3}\right)^t e^{2\pi\imath ak/n}\right)\\
&= \frac{\alpha}{n\sigma^2}\sum_{t=1}^{T}\frac{1}{t}\cos(2\pi a/n) + \frac{\alpha}{n\sigma^2}\sum_{t=1}^{T}\sum_{k=1}^{n-1}\frac{1}{t}\left(\frac{4\cos^2(\pi k/n)-1}{3}\right)^t \cos(2\pi ak/n)\\
&\le \frac{\alpha\log(T)}{n\alpha^2} + \frac{\alpha}{n\sigma^2}\sum_{k=1}^{n-1}\sum_{t=1}^{\infty}\frac{1}{t}\left(\frac{4\cos^2(\pi k/n)-1}{3}\right)^t \cos\left(\frac{2\pi(u+v-2)k}{n}\right)\\
&= \frac{\alpha\log(T)}{n\sigma^2} - \frac{\alpha}{n\sigma^2}\sum_{k=1}^{n-1}\log\left(1 - \frac{4\cos^2(\pi k/n)-1}{3}\right)\cos\left(\frac{2\pi ak}{n}\right)
\end{aligned}$$

$$\leq \frac{\alpha \log(T)}{n\sigma^2} + \frac{2\alpha}{n\sigma^2} \sum_{k=1}^{n-1} \cos\left(\frac{2\pi ak}{n}\right) \log\left(\frac{3\csc(\pi k/n)}{2}\right),$$

where csc is the cosecant function. The second equality follows from the fact that $W_{u,v}^i$ is a real number, so the imaginary part is identically zero.

In the previous bound, we give self-loops the same probability as other edges. The main reason to give self-loop a non-zero weight is to ensure irreducibility and aperiodicity of the Markov chain. The same effect can be achieved by giving any non-negligible weight to the self-loops. In particular, we can consider the following adjacency matrix:

$$\widehat{A}_R = (1-\kappa)A_R + \kappa\mathbb{I}$$

for some $\kappa > 0$. Then, the eigenvalues would be $(1-\kappa)(\omega_k + \omega^{-1}) + \kappa$. The adjacency matrix is still a circulant matrix. As a result, the eigenvectors still remain the same. Furthermore,

$$(\omega_k + \omega_k^{-1})^t \cos\left(\frac{2\pi ak}{n}\right) = 2\cos^t\left(\frac{2\pi k}{n}\right)\cos\left(\frac{2\pi ak}{n}\right) = \cos^{t-1}\left(\frac{2\pi k}{n}\right)\cos\left(\frac{2\pi(a+1)k}{n}\right).$$

Let $\kappa = \frac{1}{T^2}$. Then

$$\widehat{A}_{uv}^t \leq (1-\kappa)A_{uv}^t$$

for $t \geq 2$. Therefore, for $a = (u+v-2) \mod n$:

$$\begin{aligned}
\varepsilon_{u\to v} &\leq \frac{\alpha(1-\kappa)}{n\sigma^2}A_{uv} + \frac{\alpha(1-\kappa)}{n\sigma^2}\sum_{t=2}^{T} A_{uv}^t \\
&\leq \frac{\alpha}{n\sigma^2}A_{uv} + \frac{\alpha(1-\kappa)}{n\sigma^2}\sum_{t=2}^{T}\sum_{k=1}^{n}(\omega_k + \omega_k^{-1})^t \cos\left(\frac{2\pi ak}{n}\right) \\
&= \frac{\alpha}{n\sigma^2}A_R[u,v] + \frac{\alpha(1-\kappa)}{n\sigma^2}\sum_{t=2}^{T}\sum_{k=1}^{n}\cos^{t-1}\left(\frac{2\pi k}{n}\right)\cos\left(\frac{2\pi(a+1)k}{n}\right).
\end{aligned}$$

If $|u-v| = 1$, then we have

$$\varepsilon_{u\to v} \leq \frac{\alpha}{n\sigma^2} + \frac{\alpha(1-\kappa)}{n\sigma^2}\sum_{t=2}^{T}\sum_{k=1}^{n}\cos^{t-1}\left(\frac{2\pi k}{n}\right)\cos\left(\frac{2\pi(a+1)k}{n}\right).$$

otherwise, we have

$$\varepsilon_{u\to v} \leq \frac{\alpha(1-\kappa)}{n\sigma^2}\sum_{t=2}^{T}\sum_{k=1}^{n}\cos^{t-1}\left(\frac{2\pi k}{n}\right)\cos\left(\frac{2\pi(a+1)k}{n}\right).$$

**Star graph.** We set the vertex set of a simple star graph as

$$V = \{1, 2, \cdots, n\}$$

with the node $1$ being the central node. This gives the edge set

$$E = \{(1, i), 2 \leq i \leq n\}.$$

**Lemma D.21.** *The eigenvalues of the Laplacian of the star graph are*

$$\begin{pmatrix} 0 & 1 & \cdots & 1 & n \end{pmatrix}$$

*and the eigenvectors are $\delta_i - \delta_{i+1}$ for eigenvalues $1$ and $2 \leq i \leq n-1$. The eigenvector corresponding to eigenvalue $n$ is computed in the proof.*

*Proof.* Let $\mathbb{1}$ denote the all one vector. Then by the definition of Laplacian, $L\mathbb{1} = 0$ and eigenvalue $0$ corresponds to the eigenvector $\mathbb{1}$. Now, the trace of the Laplacian is just the sum of its eigenvalues. Therefore,

$$\mathsf{Tr}(L) = 2n - 2.$$

Let $v$ be the eigenvector for eigenvalue $n$. Then we know that

$$v \perp \mathsf{Span}\{\mathbb{1}, e_2 - e_3, e_3 - e_4, \cdots, e_{n-1} - e_n\}.$$

This implies that

$$n - 1 + v[1] = 0,$$

or that $v = \begin{pmatrix} -(n-1) & 1 & 1 & \cdots & 1 \end{pmatrix}^\top$. ∎

We can perform the spectral decomposition of the adjacency matrix of a star graph, but noting that the adjacency matrix of a star graph is

$$A = \begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

it is easy to compute the coordinates of any higher power of $A$. In particular, if $p$ is an even power, then $(u, v)$-th coordinate of $A_S^p$ is

$$A_{uv}^p = \begin{cases} (n-1)^{p/2} & u = v = 1 \\ 0 & u = 1 \text{ or } v = 1 \text{ and } u \neq v \\ (n-1)^{p/2-1} & \text{otherwise} \end{cases}$$

If $p$ is an odd power, then $(u, v)$-th coordinate of $A^p$ is

$$A_{uv}^p = \begin{cases} (n-1)^{(p-1)/2} & u = 1 \text{ or } v = 1 \\ 0 & \text{otherwise} \end{cases}$$

Since $W = \frac{A}{(n-1)}$ for a normalization constant $(n-1)$ to make $W$ doubly stochastic, we have

$$\begin{aligned} \varepsilon_{u \to v} &\leq \sum_{p=1}^{T} A_{uv}^p \frac{\alpha}{\sigma^2 p (n-1)^p} \\ &= \frac{\alpha}{\sigma^2} \left( \sum_{p \text{ is odd}}^{T} \frac{A_{uv}^p}{p(n-1)^p} + \sum_{p \text{ is even}}^{T} \frac{A_{uv}^p}{p(n-1)^p} \right). \end{aligned}$$

We now compute the privacy loss for each case. Since we only care about the privacy loss when $u \neq v$, we consider the following two cases:

1. $u = 1$ or $v = 1$ and $u \neq v$. In this case, using Proposition D.19, we have

$$\varepsilon_{u \to v} \leq \frac{\alpha}{\sigma^2} \sum_{p \text{ is odd}}^{T} \frac{(\sqrt{n-1})^{p-1}}{p(n-1)^p} \leq \frac{\alpha}{2\sigma^2 \sqrt{n-1}} \log \left( \frac{\sqrt{n-1}+1}{\sqrt{n-1}-1} \right).$$

2. $u \neq 1$ and $u \neq v$. In this case, using Proposition D.19, we have the following

$$\begin{aligned} \varepsilon_{u \to v} &\leq \frac{\alpha}{\sigma^2} \sum_{p \text{ is even}}^{T} \frac{(n-1)^{p/2-1}}{p(n-1)^p} \\ &= \frac{\alpha}{\sigma^2 (n-1)} \sum_{p \text{ is even}}^{T} \frac{(n-1)^{p/2}}{p(n-1)^p} \\ &\leq -\frac{\alpha}{\sigma^2 (n-1)} \log \left( 1 - \frac{1}{n-1} \right). \end{aligned}$$

The above calculation is when the graph is simple. To make the Markov chain aperiodic, as before, we add self-loops with a small weight on the self-loop. For example, we consider the

following adjacency matrix:

$$\widehat{A} = (1-\kappa)A + \kappa\mathbb{I}\,.$$

We pick $\kappa = \frac{1}{T^2}$, so that

$$\widehat{A}^p_{uv} \le (1-\kappa)A^p_{uv}$$

for $u \neq v$ and $p \le T$. Then if $p$ is an even power, then $(u,v)$-th coordinate of $\widehat{A}^p$ is

$$\widehat{A}^p_{uv} \le \begin{cases} (1-\kappa)(n-1)^{p/2} & u = v = 1 \\ 0 & u = 1 \text{ or } v = 1 \text{ and } u \neq v \\ (1-\kappa)(n-1)^{p/2-1} & \text{otherwise} \end{cases}$$

If $p$ is an odd power, then $(u,v)$-th coordinate of $\widehat{A}^p$ is

$$\widehat{A}^p_{uv} \le \begin{cases} (1-\kappa)(n-1)^{(p-1)/2} & u = 1 \text{ or } v = 1 \\ 0 & \text{otherwise} \end{cases}$$

Now again we have

$$\begin{aligned} \varepsilon_{u\to v} &\le \sum_{p=1}^{T} \widehat{A}^p_{uv} \frac{\alpha}{\sigma^2 p(n-1)^p} \\ &= \frac{\alpha}{\sigma^2}\left(\sum_{p \text{ is odd}}^{T} \frac{\widehat{A}^p_{uv}}{p(n-1)^p} + \sum_{p \text{ is even}}^{T} \frac{\widehat{A}^p_{uv}}{p(n-1)^p}\right). \end{aligned}$$

Using the same calculation as before, we have for all $u \neq v$,

$$\varepsilon_{u\to v} \le \begin{cases} \frac{\alpha(1-\kappa)}{2\sigma^2\sqrt{n-1}} \log\left(\frac{\sqrt{n-1}+1}{\sqrt{n-1}-1}\right) & u = 1 \text{ or } v = 1 \text{ and } u \neq v \\ -\frac{\alpha(1-\kappa)}{\sigma^2(n-1)} \log\left(1 - \frac{1}{n-1}\right) & u \neq 1 \end{cases}.$$

In particular, this means that the privacy loss for the apex node is the most. In contrast, the nodes on the arms have approximately $\sqrt{n}$ more privacy than the apex node, which is what we expect: the apex node is the only one communicating with every other node in the graph.

# Appendix E

# Proofs and Additional Results for Chapter 8

## E.1 Generic Utility Analysis of Private Fixed Point Iterations (Algorithm 8.1)

### E.1.1 Existing Result of Combettes and Pesquet [CP19]

Our convergence analysis leverages the generic convergence result of Combettes and Pesquet [CP19] for stochastic quasi-Fejér type block-coordinate fixed-point operators. Here, we briefly summarize their result (Theorem 3.1 in [CP19]) before deriving our specific analysis.

**Theorem E.1** (Mean-square convergence of stochastic quasi-Fejér type block-coordinate iterations, [CP19]). *The update rule of the stochastic quasi-Fejér type block-coordinate iterations is given by*

$$u_{k+1,b} = u_{k,b} + \rho_{k,b}\lambda_k \left(R_{k,b}\left(u_k\right) + e_{i,k} - u_{i,k}\right). \tag{E.1}$$

*Here, $b \in [B]$ denotes the b-th coordinate (block) of $u \in \mathcal{U} = \mathcal{U}_1 \times \cdots \times \mathcal{U}_B$, i.e. $u_k = [u_{k,1}, \ldots, u_{k,b}, \ldots, u_{k,B}]$, and $k$ denotes the number of iterations. We assume that the operators $(R_k)_{k\in\mathbb{N}}$ are quasi-non-expansive with common fixed point $u^*$ such that:*

$$\|R_k(u) - u^*\|^2 \leq \sum_{b=1}^{B} \tau_{k,b} \|u_b - u_b^*\|^2, \quad \forall k \in \mathbb{N}, \forall u \in \mathcal{U}, \text{ and } \exists\, \tau_{k,b} \in [0,1). \tag{E.2}$$

*Let $(\mathcal{F}_k)_{k\in\mathbb{N}}$ be a sequence of sub-sigma-algebras of $\mathcal{F}$ such that $\forall k \in \mathbb{N} : \sigma(u_0, \ldots, u_k) \subset \mathcal{F}_k \subset \mathcal{F}_{k+1}$.*

*Given this structure, we assume that the following conditions hold:*

*[a] $\inf_{k\in\mathbb{N}} \lambda_k > 0$.*

*[b] There exists a sequence of non-negative real numbers $(\alpha_k)_{k\in\mathbb{N}}$ such that $\sum_{k\in\mathbb{N}} \sqrt{\alpha_k} < +\infty$, and $\mathbb{E}\left(\|e_k\|^2 \mid \mathcal{F}_k\right) \leq \alpha_k$ for every $k \in \mathbb{N}$.*

*[c] For every $k \in \mathbb{N}, \mathcal{E}_k = \sigma\left(\rho_k\right)$ and $\mathcal{F}_k$ are independent.*

*[d] For every $b \in \{1, \ldots, B\}, \mathrm{p}_b = \mathbb{P}\left[\rho_{0,b} = 1\right] > 0$.*

*Under the assumptions [a]-[d], the iteration defined in Equation (E.2) satisfies almost surely*

$$\sum_{b=1}^{B} \omega_b \mathbb{E}\left(\|u_{k+1,b} - u_b^*\|^2 \mid \mathcal{F}_0\right) \leq \left(\prod_{j=0}^{k} \chi_j\right)\left(\sum_{b=1}^{B} \omega_b \|u_{0,b} - u_b^*\|^2\right) + \bar{\eta}_k, \quad \forall k \in \mathbb{N}. \tag{E.3}$$

*Here,*

$$\chi_k = 1 - \lambda_k\left(1 - \mu_k\right) + \sqrt{\xi_k}\lambda_k\left(1 - \lambda_k + \lambda_k\sqrt{\mu_k}\right)$$

$$\bar{\eta}_k = \sum_{j=0}^{k} \left[ \prod_{\ell=j+1}^{k} \chi_\ell \right] \lambda_j \left(1 - \lambda_j + \lambda_j \sqrt{\mu_j} + \lambda_j \sqrt{\xi_j}\right) \sqrt{\xi_j}$$
$$\xi_k = \alpha_k \max_{1 \le b \le B} \omega_b$$
$$\mu_k = 1 - \min_{1 \le b \le B} \left(p_b - \frac{\tau_{k,b}}{\omega_b}\right)$$
$$\max_{1 \le b \le B} \overline{\lim}\, \tau_{k,b} < \omega_b p_b$$

We leverage this result to derive our generic convergence analysis for the private fixed-point iteration (Algorithm 8.1), which we then instantiate to the three types of private ADMM algorithms we introduce (Section 8.4).

### E.1.2 Proof of Theorem 8.5

For ease of calculations, we mildly restrict the coordinate-wise contraction assumption made in Theorem E.1 by the following assumption of global contraction.

**Assumption E.2** (Global contraction constant)**.** *In our analysis, we assume that there exists a global contraction constant $\tau \in [0, 1)$ for the contraction operator $R_k$. Mathematically,*

$$\|R_k(u) - u^*\|^2 \le \sum_{b=1}^{B} \tau_{k,b}^2 \|u_b - u_b^*\|^2 \le \tau^2 \|u - u^*\|^2, \quad \forall k \in \mathbb{N}, \forall u \in \mathcal{U}. \tag{E.4}$$

**Theorem 8.5.** *Assume that $R$ is a $\tau$-contractive operator with fixed point $u^*$ for $\tau \in [0, 1)$. Let $P[\rho_{k,b} = 1] = q$ for some $q \in (0, 1]$. Then there exists a learning rate $\lambda_k = \lambda \in (0, 1]$ such that the iterates of Algorithm 8.1 satisfy:*

$$\mathbb{E}\left(\|u_{k+1} - u^*\|^2 \mid \mathcal{F}_0\right) \leqslant \left(1 - \frac{q^2(1-\tau)}{8}\right)^k D + 8\left(\frac{\sqrt{p}\sigma + \zeta}{\sqrt{q}\,(1-\tau)} + \frac{p\sigma^2 + \zeta^2}{q^3(1-\tau)^3}\right) \tag{E.5}$$

*where $D = \max_{u_0} \|u_0 - u^*\|^2$ is the diameter of the domain, $p$ is the dimension of $u$, $\sigma^2 > 1 - \tau$ is the variance of Gaussian noise, and $\mathbb{E}[\|e_k\|^2] \le \zeta^2$ for some $\zeta \ge 0$.*

*Proof.* We observe that Algorithm 8.1 satisfies the assumptions of Theorem E.1 if we specify $p_b = q$, $\omega_b = \frac{1}{q}$, and $\mu = 1 - q\,(1 - \tau)$. Since $\xi \ge \frac{1}{q}\mathbb{E}[\|e_k + \eta_k\|^2]$, $\mathbb{E}[\|\eta_k\|^2] \le p\sigma^2$ as zero-mean Gaussian noise are added independently to each dimension, and $\mathbb{E}[\|e_k\|^2] \le \zeta^2$, we can assign $\xi = \frac{p\sigma^2 + \zeta^2}{q}$. For ease of calculations, hereafter, we refer to $\xi$ as $\sigma_1^2$.

**Step 1: Instantiating the mean-square convergence result.** By substituting the aforementioned parameters in Equation (E.3), we obtain

$$\begin{aligned}\mathbb{E}\left(\|u_{k+1}-u^*\|^2 \mid \mathcal{F}_0\right) &\leq \chi^k \|u_0-u^*\|^2 + q\eta \\ &\leq \chi^k D + q\eta. \end{aligned} \tag{E.6}$$

Here,

$$\begin{aligned}\chi &= 1-\lambda q\left(1-\tau\right)+\lambda\sigma_1\left(1-\lambda+\lambda\sqrt{1-q\left(1-\tau\right)}\right)\\ &= 1-\lambda\left(1-b^2\right)+\lambda\sigma_1(1-\lambda+\lambda b)\\ &= 1+\lambda\left(\sigma_1-\left(1-b^2\right)\right)-\lambda^2\sigma_1(1-b),\end{aligned}$$

and

$$\begin{aligned}\eta &= \sum_{i=0}^{k}\chi^{k-i-1}\lambda\sigma\left(1+\lambda\left(\sigma_1-(1-b)\right)\right)\\ &= \frac{\frac{1}{\chi}-\chi^k}{1-\chi}\left(\chi-1+\lambda\left(1-b^2\right)+\sigma_1^2\lambda^2\right)\\ &= \left(\chi^k-\frac{1}{\chi}\right)\left(1-\frac{\sigma_1\lambda\left(\sigma_1\lambda+\frac{1-b^2}{\sigma_1}\right)}{\sigma_1\lambda\left((1-b)\lambda+\frac{1-b^2}{\sigma_1}-1\right)}\right)\\ &= \left(\chi^k-\frac{1}{\chi}\right)\left(1-\frac{\lambda\sigma_1+\frac{1-b^2}{\sigma_1}}{(1-b)\lambda+\frac{1-b^2}{\sigma_1}-1}\right).\end{aligned} \tag{E.7}$$

For simplicity, we introduce the notation $b \triangleq \sqrt{1-q\left(1-\tau\right)}$. We observe that $b \in [0,1)$ as $q \in (0,1]$ and $\tau \in [0,1)$.

**Step 2: Finding a 'good' learning rate $\lambda$.** First, we assume that there exists a $c>0$, such that the noise variance can be rewritten as $\sigma_1 = (1+c)(1-\tau)$. From Lemma E.3, we obtain that

$$\lambda \in \left(\frac{1+c-q}{(1+c)(1-b)}, \frac{1+c-q}{(1+c)(1-b)}\left(\frac{1}{2}+\frac{1}{2}\sqrt{1+4\frac{(1+c)(1-b)}{(1-\tau)(1+c-q)^2}}\right)\right).$$

For ease of further calculations, we fix $\lambda = \frac{1}{1-b}\left(1-\frac{q}{2(1+c)}\right)$. Before proceeding further, we prove that this choice of $\lambda$ belongs to the desired range. We begin by observing that

$$\frac{1-b}{1-\tau} = \frac{1-\sqrt{1-q\left(1-\tau\right)}}{(1-\tau)} \geqslant \frac{q\left(1-\tau\right)}{2(1-\tau)} = \frac{q}{2}.$$

The inequality holds due to concavity of the square root, specifically $\sqrt{1-x} \leq 1 - \frac{x}{2}$.

Thus,

$$\begin{aligned}
\frac{1+c-q}{(1+c)(1-b)}\left(\frac{1}{2}+\frac{1}{2}\sqrt{1+4\frac{(1+c)(1-b)}{(1-\tau)(1+c-q)^2}}\right) &\geq \frac{1+c-q}{(1+c)(1-b)}\left(\frac{1}{2}+\frac{1}{2}\sqrt{1+\frac{2q(1+c)}{(1+c-q)^2}}\right) \\
&= \frac{1+c-q}{(1+c)(1-b)}\left(\frac{1}{2}+\frac{1}{2}\sqrt{1+\frac{2q(1+c-q)}{(1+c-q)^2}+\frac{2q^2}{(1+c-q)^2}}\right) \\
&> \frac{1+c-q}{(1+c)(1-b)}\left(\frac{1}{2}+\frac{1}{2}\sqrt{1+\frac{2q}{(1+c-q)}+\frac{q^2}{(1+c-q)^2}}\right) \\
&= \frac{1+c-q}{(1+c)(1-b)}\left(1+\frac{q}{2(1+c-q)}\right) \\
&= \frac{1}{1-b}\left(1-\frac{q}{2(1+c)}\right).
\end{aligned}$$

**Step 3: Understanding the impact of the noise term $\eta$.** First, we investigate the term $\mathrm{A} \triangleq \frac{\lambda\sigma_1+\frac{1-b^2}{\sigma_1}}{(1-b)\lambda+\frac{1-b^2}{\sigma_1}-1}$ in (E.7).

$$\begin{aligned}
\text{Denominator of A} &= (1-b)\lambda + \frac{1-b^2}{\sigma_1} - 1 \\
&= 1 - \frac{q}{2(1+c)} + \frac{q\,(1-\tau)}{(1+c)\,(1-\tau)} - 1 \\
&= \frac{q}{2(1+c)} \\
\text{Numerator of A} &= \lambda\sigma_1 + \frac{1-b^2}{\sigma_1} \\
&= \frac{(1+c)\,(1-\tau)}{1-b}\left(1-\frac{q}{2(1+c)}\right) + \frac{q}{1+c} \\
&= \frac{(1+c)\,(1+b)}{q}\left(1-\frac{q}{2(1+c)}\right) + \frac{q}{1+c} \\
&= \frac{(1+c)\,(1+b)}{q} + \frac{q}{1+c}\left(1-\frac{(1+c)\,(1+b)}{2q}\right)
\end{aligned}$$

Thus, we get

$$A = \frac{2(1+b)(1+c)^2}{q^2} + 2 - \frac{(1+c)\,(1+b)}{q}$$

and,

$$1 - A = \frac{(1+c)\,(1+b)}{q} - 1 - \frac{2(1+b)(1+c)^2}{q^2}.$$

By substituting $(1-A)$ in Equation (E.7) and plugging back $\eta$ into Equation (E.6), we get

$$\begin{aligned}
\mathbb{E}\left(\|u_{k+1}-u^*\|^2 \mid \mathcal{F}_0\right) &\leq \chi^k D + q\left(\chi^k - \frac{1}{\chi}\right)(1-A) \\
&= \chi^k(D+q(1-A)) - \frac{1}{\chi}q(1-A) \\
&= \chi^k\left(D+(1+c)(1+b)-q-\frac{2(1+b)(1+c)^2}{q}\right) \\
&\quad + \frac{1}{\chi}\left(-(1+c)(1+b)+q+\frac{2(1+b)(1+c)^2}{q}\right) \\
&\leq \chi^k\left(D+(1+c)(1+b)\right) + \frac{1}{\chi}\left(q+\frac{2(1+b)(1+c)^2}{q}\right) \\
&\leq \chi^k\left(D+2(1+c)\right) + \frac{1}{\chi}\left(q+\frac{2(1+b)(1+c)^2}{q}\right). \qquad \text{(E.8)}
\end{aligned}$$

**Step 4: Upper & lower bounding $\chi$.** We can rewrite $\chi$ as follows:

$$\begin{aligned}
\chi &= 1+\lambda\left(\sigma_1-\left(1-b^2\right)\right)-\lambda^2\sigma_1(1-b) \\
&= 1+\lambda\left((1+c)(1-\tau)-q(1-\tau)\right)-\lambda^2(1+c)(1-\tau)(1-b) \\
&= 1+\lambda(1-\tau)(1+c-q)-\lambda^2(1+c)(1-\tau)(1-b) \\
&= 1+\frac{(1-\tau)(1+c-q)(1+c-q/2)}{(1+c)(1-b)}-\frac{(1-\tau)(1+c-q/2)^2}{(1+c)(1-b)} \\
&= 1-\frac{q(1-\tau)(1+c-q/2)}{2(1-b)(1+c)} \\
&= 1-\frac{(1+b)(1+c-q/2)}{2(1+c)}
\end{aligned}$$

*Lower bound:*

$$\chi = 1-\frac{1+b}{2}+\frac{q(1+b)}{4(1+c)} > \frac{1-b}{2} = \frac{1-\sqrt{1-q(1-\tau)}}{2} \geq \frac{q(1-\tau)}{4}$$

The inequality holds due to the fact that $b=\sqrt{1-q(1-\tau)} < 1-\frac{q(1-\tau)}{2}$.

*Upper bound:*

$$\begin{aligned}
\chi = 1-\frac{(1+b)}{2}\left(1-\frac{q}{2(1+c)}\right) = \frac{1-b}{2}+\frac{q(1+b)}{4(1+c)} &\leqslant \frac{1}{2}+\frac{q(1+b)}{4} \\
&\leqslant \frac{1}{2}+\frac{q}{4}\left(2-\frac{q(1-\tau)}{2}\right) \\
&\leqslant 1-\frac{q^2(1-\tau)}{8}.
\end{aligned}$$

The first inequality holds for any non-negative $b, c, q$. The second inequality leverages the fact that $b = \sqrt{1-q(1-\tau)} \leq 1 - \frac{q(1-\tau)}{2}$. The final inequality follows from the fact that $q \in (0, 1]$.

**Step 5: Final touch.** By substituting upper and lower bounds of $\chi$ in Equation (E.8), we get

$$\begin{aligned}
\mathbb{E}\left(\|u_{k+1} - u^*\|^2 \mid \mathcal{F}_0\right) &< \left(1 - \frac{q^2(1-\tau)}{8}\right)^k (D + 2(1+c)) + \frac{4}{q(1-\tau)}\left(q + \frac{2(1+b)(1+c)^2}{q}\right) \\
&= \left(1 - \frac{q^2(1-\tau)}{8}\right)^k \left(D + \frac{2\sigma_1}{(1-\tau)}\right) + \frac{4}{q\,(1-\tau)}\left(q + \frac{2(1+b)\sigma_1^2}{q(1-\tau)^2}\right) \\
&= \left(1 - \frac{q^2(1-\tau)}{8}\right)^k \left(D + \frac{2\sigma_1}{(1-\tau)}\right) + \frac{4}{(1-\tau)} + \frac{8(1+b)\sigma_1^2}{q^2(1-\tau)^3} \\
&\leq \left(1 - \frac{q^2(1-\tau)}{8}\right)^k \left(D + \frac{2\sigma_1}{(1-\tau)}\right) + \left(\frac{4}{(1-\tau)} + \frac{16\sigma_1^2}{q^2(1-\tau)^3}\right) \\
&\leq \left(1 - \frac{q^2(1-\tau)}{8}\right)^k \left(D + \frac{2(\sigma\sqrt{p}+\zeta)}{\sqrt{q}\,(1-\tau)}\right) + \left(\frac{4}{(1-\tau)} + \frac{8(p\sigma^2+\zeta^2)}{q^3(1-\tau)^3}\right) \\
&\leq \left(1 - \frac{q^2(1-\tau)}{8}\right)^k D + \left(\frac{8(\sigma\sqrt{p}+\zeta)}{\sqrt{q}\,(1-\tau)} + \frac{8(p\sigma^2+\zeta^2)}{q^3(1-\tau)^3}\right)
\end{aligned}$$

We can also alternatively write the result as follows. Since $(1-a)^k \leq \exp(-ak)$ for $a \in [0, 1)$ and $k \in \mathbb{N}$, we have:

$$\mathbb{E}\left(\|u_{k+1} - u^*\|^2 \mid \mathcal{F}_0\right) \leqslant \exp\left(-\frac{q^2(1-\tau)}{8}k\right) D + \left(\frac{8(\sigma\sqrt{p}+\zeta)}{\sqrt{q}\,(1-\tau)} + \frac{8(p\sigma^2+\zeta^2)}{q^3(1-\tau)^3}\right)$$

■

### E.1.3 Technical Lemma on the Learning Rate

We prove below a technical lemma used in the proof of Theorem 8.5.

**Lemma E.3** (Choices of the Learning Rate). *In order to ensure convergence of Algorithm 8.1, we should choose the learning rate $\lambda$ in the range*

$$\left(\frac{1+c-q}{(1+c)(1-b)}, \frac{1+c-q}{(1+c)(1-b)}\left(\frac{1}{2} + \frac{1}{2}\sqrt{1 + 4\frac{(1+c)(1-b)}{(1-\tau)(1+c-q)^2}}\right)\right).$$

*Here, we assume that there exists $c > 0$ such that $\sigma_1 \triangleq \frac{\sigma\sqrt{p}+\zeta}{\sqrt{q}} \triangleq (1+c)(1-\tau)$, $b \triangleq \sqrt{1-q(1-\tau)}$, $\sigma$ is the noise variance, $\tau \in [0, 1)$ is the contraction factor, and $q \in (0, 1]$.*

*Proof.* In order to ensure convergence of the algorithm, we need to satisfy $0 < \chi < 1$. We observe that

$$\chi = 1 + \lambda\left(\sigma_1 - \left(1 - b^2\right)\right) - \lambda^2\sigma_1(1-b),$$

As $\chi$ is a function of the learning rate, the upper and lower bounds on $\chi$ impose lower and upper bounds on the desired learning rate $\lambda$.

**Step 1: Lower Bounding $\lambda$.**

$$\begin{aligned}
\chi < 1 &\implies 1 + \lambda\left(\sigma_1 - \left(1 - b^2\right)\right) - \lambda^2\sigma_1(1-b) < 1 \\
&\underset{(a)}{\implies} \left(\sigma_1 - \left(1 - b^2\right)\right) - \lambda\sigma_1(1-b) < 0 \\
&\implies \frac{\sigma_1 - (1-b^2)}{\sigma_1(1-b)} < \lambda \\
&\implies \frac{(1+c)(1-\tau) - q\left(1-\tau^2\right)}{(1+c)(1-\tau)(1-b)} < \lambda \\
&\implies \frac{1+c-q}{(1+c)(1-b)} < \lambda
\end{aligned}$$

Step (a) holds true for $\lambda > 0$, i.e. for any positive learning rate.

**Step 2: Upper Bounding $\lambda$.** As $\chi > 0$, we should choose the learning rate $\lambda$ in a range such that the following quadratic equation satisfies

$$1 + \lambda\left(\sigma_1 - \left(1 - b^2\right)\right) - \lambda^2\sigma_1(1-b) > 0.$$

Since the coefficient corresponding to $\lambda^2$ is negative, the quadratic equation stays positive only between its two roots:

$$\lambda_{inf} = \frac{(\sigma_1 - (1-b^2)) - \sqrt{(\sigma_1 - (1-b^2))^2 + 4\sigma_1(1-b)}}{2\sigma_1(1-b)},$$

and

$$\lambda_{sup} = \frac{(\sigma_1 - (1-b^2)) + \sqrt{(\sigma_1 - (1-b^2))^2 + 4\sigma_1(1-b)}}{2\sigma_1(1-b)}.$$

Since the smallest root $\lambda_{inf}$ is negative, and we care about only positive learning rates, it provides a vacuous bound. Thus, we can ignore it.

Thus, we conclude that

$$\begin{aligned}
\lambda &< \frac{(\sigma_1 - (1-b^2)) + \sqrt{(\sigma_1 - (1-b^2))^2 + 4\sigma_1(1-b)}}{2\sigma_1(1-b)} \\
&= \frac{(1+c-q)(1-\tau) + \sqrt{(1+c-q)^2(1-\tau)^2 + 4(1+c)(1-b)(1-\tau)}}{2(1+c)(1-\tau)(1-b)}
\end{aligned}$$

$$= \frac{(1+c-q)+(1+c-q)\sqrt{1+4\frac{(1+c)(1-b)}{(1-\tau)(1+c-q)^2}}}{2(1+c)(1-b)}$$
$$= \frac{1+c-q}{(1+c)(1-b)}\left(\frac{1}{2}+\frac{1}{2}\sqrt{1+4\frac{(1+c)(1-b)}{(1-\tau)(1+c-q)^2}}\right)$$

to obtain a valid convergence of the algorithm. ■

## E.2 Derivation of Private ADMM Updates

In this section, we give details on how to obtain the private ADMM updates given in Algorithm 8.2, Algorithm 8.4 and Algorithm 8.3 from our general noisy fixed-point iteration (Algorithm 8.1).

### E.2.1 Warm-up: Non-Private ADMM

For clarity and self-completeness, we start by deriving the standard ADMM updates from the fixed-point iteration formulation described in Section 8.2.2. This derivation follows the lines of [GFB16, Appendix B therein].

Recall that ADMM solves an optimization problem of the form (8.3), which we restate here for convenience:

$$\begin{array}{ll} \min_{x,z} & f(x) + g(z) \\ s.t. & Ax + Bz = c \end{array} \tag{E.9}$$

We also recall the definition of the infimal postcomposition.

**Definition E.4** (Infimal postcomposition)**.** *Let $M$ be a linear operator. The infimal postcomposition $M \triangleright f$ is defined by*

$$(M \triangleright f)(y) = \inf\{f(x) \mid Mx = y\}.$$

As mentioned in Section 8.2.2, the minimization problem above can be rewritten as

$$\min_u (-A \triangleright f)(-u - c) + (-B \triangleright g)(u).$$

Introducing $p_1(u) = (-A \triangleright f)(-u - c)$ and $p_2(u) = (-B \triangleright g)(u)$ recovers a minimization problem solvable with the Douglas-Rachford algorithm. Formally, the $\lambda$-averaged ADMM can be written as the following fixed-point operator:

$$u_{k+1} = u_k + \lambda \left( R_{\gamma p_1}(R_{\gamma p_2}(u_k)) - u^k \right), \tag{E.10}$$

where $R_{\gamma p_1} = 2\,\mathrm{prox}_{\gamma p_1} - I$ and $R_{\gamma p_2} = 2\,\mathrm{prox}_{\gamma p_2} - I$.

From this generic formula, we can recover the standard ADMM updates in terms of $x$ and $z$. We start by rewriting $R_{\gamma p_2}(u)$:

$$\begin{aligned} R_{\gamma p_2}(u) &= 2\,\mathrm{prox}_{\gamma p_2}(u) - u \\ &= 2 \arg\min_v \left\{ \inf_z \{g(z) \mid -Bz = v\} + \frac{1}{2\gamma}\|u - v\|^2 \right\} - u \\ &= -2B \arg\min_z \left\{ g(z) + \frac{1}{2\gamma}\|Bz + u\|^2 \right\} - u. \end{aligned}$$

This leads to the introduction of the $z$ variable with associated update:

$$z_{k+1} = \arg\min_z \left\{ g(z) + \frac{1}{2\gamma}\|Bz + u_k\|^2 \right\}. \tag{E.11}$$

Similarly, we can rewrite $R_{\gamma p_1}$:

$$\begin{aligned} R_{\gamma p_1}(u) &= 2\,\mathrm{prox}_{\gamma p_1}(u) - u \\ &= 2 \arg\min_v \left\{ \inf_x \{f(x) \mid -Ax = -v - c\} + \frac{1}{2\gamma}\|u - v\|^2 \right\} - u \\ &= 2A \arg\min_x \left\{ f(x) + \frac{1}{2\gamma}\|Ax - u - c\|^2 \right\} - 2c - u, \end{aligned}$$

which leads to the introduction of the $x$ variable with associated update:

$$x_{k+1} = \arg\min_x \left\{ f(x) + \frac{1}{2\gamma}\|Ax + 2Bz_{k+1} + u_k - c\|^2 \right\}. \tag{E.12}$$

Based on (E.11) and (E.12), we can rewrite:

$$\begin{aligned} R_{\gamma p_1} R_{\gamma p_2}(u_k) &= R_{\gamma p_1}(-2Bz_{k+1} - u_k) \\ &= 2Ax_{k+1} - 2c - (-2Bz_{k+1} - u_k) \\ &= 2(Ax_{k+1} + Bz_{k+1} - c) + u_k, \end{aligned}$$

which in turns gives for the update of variable $u$ in (E.10):

$$u_{k+1} = u_k + 2\lambda(Ax_{k+1} + Bz_{k+1} - c), \tag{E.13}$$

The updates (E.11), (E.12) and (E.13) correspond to the standard ADMM updates [Boy+11; GFB16].

**Algorithm E.1:** General Private ADMM to solve problem (E.9)

**Input:** initial point $u_0$, step size $\lambda \in (0, 1]$, privacy noise variance $\sigma^2 \geq 0$, Lagrange parameter $\gamma > 0$
**for** $k = 0$ *to* $K - 1$ **do**
  $z_{k+1} = \arg\min_z \left\{ g(z) + \frac{1}{2\gamma} \|Bz + u_k\|^2 \right\}$
  $x_{k+1} = \arg\min_x \left\{ f(x; \mathcal{D}) + \frac{1}{2\gamma} \|Ax + 2Bz_{k+1} + u_k - c\|^2 \right\}$
  $u_{k+1} = u_k + 2\lambda \left( Ax_{k+1} + Bz_{k+1} - c + \frac{1}{2}\eta_{k+1} \right)$ with $\eta_{k+1} \sim \mathcal{N}\left(0, \sigma^2 \mathbb{I}\right)$
**return** $z^K$

### E.2.2 General Private ADMM

We now introduce a general private version of ADMM to solve problem (E.9). In this generic part, we consider without loss of generality that the data-dependent part is in the function $f$. For clarity, we denote $f(x)$ by $f(x; \mathcal{D})$ to make the dependence on the dataset $\mathcal{D}$ explicit.

Following our general noisy fixed-point iteration (Algorithm 8.1), the private counterpart of the non-private ADMM iteration (E.10) is given by:

$$u_{k+1} = u_k + \lambda \left( R_{\gamma p_1}(R_{\gamma p_2}(u_k); \mathcal{D}) - u_k + \eta_{k+1} \right),$$

where the notation $R_{\gamma p_1}(\cdot; D)$ is again to underline the data-dependent part of the computation.

By following the same derivations as in Appendix E.2.1, we obtain the following equivalent update:

$$u_{k+1} = u_k + 2\lambda \left( Ax_{k+1} + Bz_{k+1} - c + \frac{1}{2}\eta_{k+1} \right),$$

where $z_{k+1}$ and $x_{k+1}$ are defined as in (E.11) and (E.12) respectively. The full algorithm is given in Algorithm E.1. Note that we return only $z_K$, which is differentially private by postprocessing of $u_{K-1}$ (see Appendix E.3). In contrast, returning $x_K$ would violate differential privacy as the last update interacts with the data without subsequent random perturbation. In many problems (such as the consensus problem considered below), returning $z_K$ is sufficient for all practical purposes. Note that when $A$ is invertible (which is the case for consensus, see below), one can recover from $z_K$ the unique $\tilde{x}_K = A^{-1}(c - Bz_K)$ such that $(\tilde{x}_K, z_K)$ satisfies the constraint in problem (E.9).

### E.2.3 Instantiations for the Consensus Problem

We now instantiate the generic private ADMM update given in Appendix E.2.2 to the consensus problem and derive centralized, fully decentralized and federated private ADMM algorithms for ERM.

Recall that the ERM problem (8.5) can be reformulated as the consensus problem (8.6), which we restate below for convenience:

$$\begin{aligned}&\min_{x\in\mathbb{R}^{np},z\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n} f\left(x_i;d_i\right)+r(z)\\&\text{s.t } x_i = z \quad \forall i,\end{aligned}$$

which is a special case of problem (E.9) with $x=(x_1,\dots,x_n)^\top$ composed of $n$ blocks of $p$ coordinates, $f(x)=f(x;\mathcal{D})=\frac{1}{n}\sum_{i=1}^n f(x_i;d_i)$, $g(z)=r(z)$, $c=0$, $A=I$ and $B=-I_{n(p\times p)}\in\mathbb{R}^{n\times p}$ where $I_{n(p\times p)}\in\mathbb{R}^{n\times p}$ denotes $n$ stacked identity matrices of size $p\times p$.

**Centralized private ADMM (Algorithm 8.2).** We use the specific structure of the consensus problem to simply the general private ADMM updates in Appendix E.2.2. The $z$-update gives:

$$\begin{aligned}z_{k+1}&=\arg\min_z\left\{r(z)+\frac{1}{2\gamma}\left\|\begin{pmatrix}I\\\dots\\I\end{pmatrix}z-u_k\right\|^2\right\},\\z_{k+1}&=\operatorname{prox}_{\gamma r}\left(\frac{1}{n}\sum_{i=1}^n u_{k,i}\right).\end{aligned}$$

For the $x$-update, we have:

$$x_{k+1}=\arg\min_x\left\{f(x;\mathcal{D})+\frac{1}{2\gamma}\left\|x-2\begin{pmatrix}I\\\dots\\I\end{pmatrix}z_{k+1}+u_k\right\|^2\right\}.$$

As $f$ is fully separable, this can be decomposed into $n$ block-wise updates as:

$$\begin{aligned}x_{k+1,i}&=\arg\min_{x_i}\left\{f(x_i;d_i)+\frac{1}{2\gamma}\left\|x_i-2z_{k+1}+u_{k,i}\right\|\right\}\\&=\operatorname{prox}_{\gamma f_i}(2z_{k+1}-u_{k,i}).\end{aligned}$$

Finally, the $u$-update writes:

$$u_{k+1}=u_k+2\lambda\left(x_{k+1}-\begin{pmatrix}I\\\dots\\I\end{pmatrix}z_{k+1}+\frac{1}{2}\eta_{k+1}\right),$$

which can be equivalently written as block-wise updates:

$$u_{k+1,i} = u_{k,i} + 2\lambda \left( x_{k+1,i} - z_{k+1} + \frac{1}{2}\eta_{k+1,i} \right).$$

Algorithm 8.2 shows the resulting algorithm when cycling over the $n$ blocks of $x$ and $u$ in lexical order (which is equivalent to considering a single block, i.e., $B = 1$). But remarkably, the flexibility of our general noisy fixed-point iteration (Algorithm 8.1) and associated utility result (Theorem 8.5) allows us to cover many other interesting cases, some of which directly leading to federated and fully decentralized learning algorithms (see below). In particular, we can sample the blocks in a variety of ways, such as:

1. cycling over an independently chosen random permutation of the blocks at each iteration (the corresponding utility can be obtained by setting $q = 1$ in Theorem 8.5);
2. choosing a single random block at each iteration $k$ (this is used to obtained our fully decentralized algorithm);
3. choosing a random subset of $m$ blocks (this is used to obtain our federated algorithm with user sampling).

The utility guarantees can be obtained from Theorem 8.5 by setting $q = 1$ in case 1, $q = 1/n$ in case 2, and $q = m$ in case 3.

**Federated private ADMM (Algorithm 8.3).** Our federated private ADMM algorithm exactly mimics the updates of centralized private ADMM (Algorithm 8.2), which can be executed in a federated fashion since (i) the blocks $x_i$ and $u_i$ associated to each user $i$ can be updated in parallel by each user, and (ii) if each user $i$ shares $u_{k+1,i} - u_{k,i}$ with the server, then the latter can execute the rest of the updates. The more general version with user sampling given in Algorithm 8.3 is obtained by choosing a random subset of $m$ blocks (users) uniformly at random.

**Fully decentralized private ADMM (Algorithm 8.4).** In the fully decentralized setting, each user $i$ with local dataset $d_i$ is associated with blocks $x_i$ and $u_i$. Our fully decentralized private ADMM algorithm (Algorithm 8.4) directly follows from a block-wise version of Algorithm 8.2, where at each iteration $k$ we select uniformly at random a single block (user) to update. This corresponds to a user performing an update on its local parameters before sending it to another user chosen at random.

## E.3 Privacy Analysis of our ADMM Algorithms

**Privacy amplification by subsampling**

When a DP algorithm is executed on a random subsample of data points, and the choice of this subsampling remains secret, we can obtain privacy amplification. This privacy amplification by subsampling effect has been extensively studied under various sampling schemes [BBG18; MTZ19] and is classically used in the privacy analysis of DP-SGD [BST14; Aba+16; AT22]. While tighter bounds can be computed numerically, here for the sake of simplicity we use a simple closed-form expression which gives the order of magnitude of the amplification.

**Lemma E.5** (Amplification by subsampling, [AT22])**.** *Let $q < 1/5$, $\alpha > 1$ and $\sigma \geq 4$. Then, for $\alpha \leq \left(M^2\sigma^2/2 - \log\left(5\sigma^2\right)\right) / \left(M + \log(q\alpha) + 1/\left(2\sigma^2\right)\right)$ where $M = \log(1 + 1/(q(\alpha-1)))$, the subsampled Gaussian mechanism with probability $q$ and noise parameter $\sigma^2$ satisfies $(\alpha, \varepsilon_{samp})$-RDP with*

$$\varepsilon_{samp} \leq \frac{2\alpha q^2 \Delta^2}{\sigma^2}.$$

### E.3.1 Sensitivity Bounds

We aim at bounding the privacy loss of the general centralized ADMM introduced in Section E.2.1. We assume that $K$ iterations are done with only $f$ interacting with data, i.e., the data-dependent step lies in the $x$-update. We assume that all data points are used with uniform weighting, meaning that $f$ can be written as $f(x; \mathcal{D}) = \frac{1}{n}\sum_{i=1}^{n} f(x; d_i)$.

To bound the privacy loss, we aim at computing the Rényi divergence between the distribution of the outputs, which can be linked to the sensitivity of the fixed-point update (E.13) to the change of one data point. For any pair of neighboring datasets $\mathcal{D} \sim \mathcal{D}'$ that differs only on data item $d_i$ (i.e., $d_j \neq d'_j \implies i = j$) and any $u$, we thus want to bound the difference between $T(u)$ computed on dataset $\mathcal{D}$ and $T'(u)$ computed on the dataset $\mathcal{D}'$. We note $x$ (resp. $x'$) and $z$ (resp. $z'$) the primal variables in the calculation.

We first investigate how the sensitivity of the data-dependent update propagates to $u$. As only $x$-updates are data-dependent, the $z$ stays identical for $\mathcal{D}$ and $\mathcal{D}'$ and thus we have:

$$T(u) - T'(u) = 2\lambda A(x - x'). \tag{E.14}$$

We bound the sensitivity by assuming that $A$ has its smallest singular value $\omega_A > 0$.

Let us define $\varphi(x) = \frac{1}{2\gamma} \|Ax + Bz + u + c\|^2$. $\varphi$ is twice differentiable and we have

$$\begin{aligned}
\nabla_x \varphi(x) &= \frac{1}{2\gamma} \nabla_x \left( x^\top A^\top A x - 2(Bz + u + c)^\top Ax + (Bz + u + c)^\top (Bz + u + c) \right) \\
&= \frac{1}{2\gamma} \left( 2A^\top Ax - 2A(Bz + u + c) \right), \\
\nabla_x^2 \varphi(x) &= \frac{1}{\gamma} A^\top A.
\end{aligned}$$

Thus, $\varphi$ is $\mu$-strongly convex if and only:

$$\mu I_n \preceq \frac{1}{\gamma} A^T A.$$

This is satisfied when the smallest eigenvalue of $A^\top A$ is larger than $\mu$. This corresponds to the same condition on the smallest singular value $\omega_A$ of $A$, hence

$$\omega_A \geq \mu\gamma,$$

and thus $\varphi$ is $\frac{\omega_A}{\gamma}$-strongly convex.

Let us now consider $F(x) = f(x; \mathcal{D}) + \varphi(x)$ and $F'(x) = f'(x; \mathcal{D}') + \varphi(x)$. We assume that $f_i(\cdot) = f(\cdot; d_i)$ are convex, differentiable and $L$-Lipschitz with respect to the $l_2$ norm for all possible $d$. Then, using a classic result on the sensitivity of the $\arg\min$ of strongly convex functions [CMS11], the sensitivity of $\arg\min F(x)$ is bounded by:

$$\|x - x'\| \leqslant \frac{2L\gamma}{n\omega_A}.$$

Finally, by re-injecting this formula into (E.14), we get the final bound:

$$\|T(u) - T'(u)\| \leq \frac{4\lambda L\gamma \|A\|_2}{n\omega_A}. \tag{E.15}$$

**Special case of the consensus problem.** In the case of the consensus problem, we can derive a tighter upper bound for the sensitivity of the block-wise update for which the data point is different between $\mathcal{D}$ and $\mathcal{D}'$:

$$T(u)_i - T'(u)_i = 2\lambda(x_i - x_i').$$

In this case, the $x_i$ can be simply rewritten as $\mathrm{prox}_{\gamma f_i}(2z - u)$, where $f_i$ is $L$-Lipschitz, and we have:

$$\|x_i - x_i'\| \leq 2L\gamma.$$

Therefore, $\|T(u)_i - T'(u)_i\| \leq 4\lambda L\gamma$ and then

$$\|T(u) - T'(u)\| \leq \frac{4\lambda L\gamma}{n}. \tag{E.16}$$

### E.3.2 General Centralized Private ADMM

We can now derive the privacy loss of our general private ADMM algorithm (Algorithm E.1).

**Theorem E.6** (Private classic centralized ADMM)**.** *Let $A$ be full rank and $\omega_A > 0$ the minimal module of its singular values. After performing $K$ iterations, Algorithm E.1 is $(\alpha, \varepsilon(\alpha))$-RDP with*

$$\varepsilon(\alpha) = \frac{8K\alpha \|A\|_2^2 L^2\gamma^2}{\sigma^2 n^2 \omega_A^2}. \tag{E.17}$$

*Proof.* Recall that the output of the algorithm is $z_K$. We also recall that, for a function of sensitivity $\Delta$, we know that the addition of Gaussian noise of parameter $\sigma^2$ gives $(\alpha, \alpha\frac{\Delta^2}{2\sigma^2})$-RDP.

Hence, using the sensitivity bound given in (E.15), a single update leads to a privacy loss of

$$\varepsilon(\alpha) = \frac{8\alpha \|A\|_2^2 L^2\gamma^2}{\sigma^2 n^2 \omega_A^2}.$$

We conclude by using by the composition property of RDP over the $K$ iterations and the robustness to postprocessing. ■

Note that the theorem only requires the matrix $A$ to be full rank, which is a mild assumption.

In particular, for the consensus problem, $A$ is the identity matrix. This leads to the following privacy guarantee.

**Theorem E.7.** *After performing $K$ iterations, Algorithm 8.2 is $(\alpha, \varepsilon)$-RDP with*

$$\varepsilon(\alpha) = \frac{8K\alpha L^2\gamma^2}{\sigma^2 n^2}.$$

*Proof.* The proof is the same as that of Theorem E.6 except that we use the improved sensitivity bound given in (E.16). ■

### E.3.3 Federated Private ADMM with Subsampling

As explained in the main text, we can derive two levels of privacy for the federated algorithm. One is achieved at the level of users thanks their local injection of noise: this ensures local DP. The second one in achieved with respect to a third party observing only the final model: this is central DP. In the latter case, the local privacy level is amplified by the subsampling of users and the sensitivity is further reduced by the aggregation step.

We start by the local privacy guarantee.

**Theorem E.8** (LDP of federated ADMM)**.** *Let $K_i$ be the number of participations of user $i$. Algorithm 8.3 satisfies $(\alpha, \varepsilon_i)$ local RDP for user $i$ with*

$$\varepsilon_i \leq \mathcal{O}\left(\frac{8K_i\alpha L^2\gamma^2}{\sigma^2}\right).$$

*Proof.* We first derive the local privacy loss of sharing $z$. Using the sensitivity bound (E.16) derived for the centralized case and the fact that we consider a post-processing of $u$, we have

$$\varepsilon_{loc} \leq \frac{8\alpha L^2\gamma^2}{\sigma^2}. \tag{E.18}$$

We obtain the total local privacy loss by composition over the $K_i$ participations of user $i$. ■

We now turn to the central privacy guarantee.

**Theorem E.9.** *Let $m < n/5$, $\alpha > 1$ and $\sigma \geq 4$, then for $\alpha \leq \frac{M^2\sigma^2/2 - \log\left(5\sigma^2\right)}{M + \log(m\alpha/n) + 1/(2\sigma^2)}$ where $M = \log(1 + 1/(m(\alpha-1)/n))$. Then, Algorithm 8.3 has for central DP loss the following bound:*

$$\varepsilon \leq \frac{16K\alpha L^2\gamma^2}{n^2\sigma^2}$$

*Proof.* Recall that we subsample $m$ participants at each round. By the reduction of sensitivity due to the aggregation of the $m$ participations, the initial privacy loss for one iteration is $\varepsilon_{loc}/m^2$, where $\varepsilon_{loc}$ is given in (E.18). Then, applying privacy amplification by subsampling (see Appendix E.3) of $m$ users among $n$ leads to

$$\varepsilon \leq \frac{8\alpha L^2\gamma^2}{m^2\sigma^2}\frac{2m^2}{n^2}.$$

We conclude using composition over the $K$ rounds of the algorithm. ■

### E.3.4 Fully Decentralized Private ADMM

In the fully decentralized setting, the local privacy loss is the same as in the previous section for the federated case. However, the threat model is quite different. The privacy guarantees are with respect to the other users' view, and each user will only observe information in time steps where he/she participates.

We characterize the privacy loss by decomposing the problem as follows. Starting from the LDP loss, we derive the privacy loss suffered by a user $i$ when the $z$ variable is observed $m$ steps after the contribution made by $i$. This is similar to the classic setting of privacy amplification by iteration where a model is only available after a given number of steps (see Theorem 3.11). Then, from the formula for a fixed number of steps, we derive the privacy loss that accounts for the secrecy of the path and the randomness of its length. This is done by using the weak convexity property of the Rényi divergence [Fel+18] to weight each scenario according to the probability of the possible lengths. These probabilities can be easily computed as we consider a complete graph for the communication graph. We conclude the proof by using composition over the maximum number of times $K_i$ any user participates to the computation.

For convenience, we first restate the theorem, and then give the full proof.

**Theorem 8.12.** *Assume that the loss function $f(\cdot, d)$ is L-Lipschitz for any local dataset $d$ and consider user-level DP. Let $\alpha > 1, \sigma > 2L\gamma\sqrt{\alpha(\alpha-1)}$ and $K_i$ the maximum number of contribution of a user. Then Algorithm 8.4 satisfies $(\alpha, \frac{8\alpha K_i L^2\gamma^2 \log n}{\sigma^2 n})$-network RDP.*

*Proof.* Here, a given user $j$ can only infer information about the other users when it participates, by observing the current value of the $z$ variable. Therefore, we can write the view of user $j$ as:

$$\mathcal{O}_j(\mathcal{A}(D)) = (z_{k_l(j)})_{l=1}^{K_j},$$

where $k_l(j)$ is the time of $l$-th contribution of user $j$ to the computation, and $K_j$ is the total number of times that $j$ contributed during the execution of algorithm. As we consider the complete graph, the probability to visit $j$ at any step is exactly $1/n$. Hence, we have closed forms for the probability that the random walk goes from a user $i$ to another user $j$ in $m$ steps. Specifically, it follows the geometric law of parameter $1/n$.

As an intermediate step of the proof, we thus express the privacy loss induced by a user $i$ with respect to another user $j$ when there is exactly $m$ steps after the participation of $i$ to reach $j$, meaning than $j$ will only observe the variable $z_{k+m}$ if $i$ participated at time $k$, and thus the contribution of $i$ has already been mixed with $m$ subsequent steps of the algorithm.

In this case, the privacy loss can be computed from the local privacy loss $\varepsilon_{loc}$ in Equation (E.18), and the use of privacy amplification by iteration in Theorem 3.11 where we have

$s_1 = \varepsilon_{loc}$ and $s_{i>1} = 0$, and we set $a_i = \varepsilon_{loc}/m$. This leads to following bound:

$$\varepsilon \leq \sum_{i=1}^{m} D_\alpha(\mathcal{N}\left(0, \sigma^2\right) ||\mathcal{N}(a_i, \sigma^2)) \leq \frac{8\alpha L^2\gamma^2}{\sigma^2 m}.$$

Now that we have a bound for a fixed number of steps between the two users, we can compute the privacy loss for the random walk. Using the fact that the walk remains private to the users, i.e. they do not observe the trajectory of the walk except the times it passed through them, we can apply the weak convexity of the Rényi divergence [Fel+18].

Let us fix a contribution of user $i$ at some time $k(i)$. We apply this lemma to $\rho$ the distribution of the number of steps before reaching user $j$, which follows a geometric law of parameter $1/n$. This gives:

$$\begin{aligned} D_\alpha(z_j||z'_j) &\leq \sum_{k=1}^{K-k(i)} \frac{1}{n}(1-\frac{1}{n})^k \frac{8\alpha L^2\gamma^2}{2\sigma^2 k} \\ &\leq \frac{8\alpha L^2\gamma^2}{\sigma^2 n} \sum_{k=1}^{\infty} \frac{(1-1/n)^k}{k} \\ &\leq \frac{8\alpha L^2\gamma^2 \log n}{\sigma^2 n}. \end{aligned}$$

Finally, we use composition to bound the total privacy loss. Each user participates $K/n$ times in average, and this estimate concentrates as $K$ increases. For the sake of simplicity, we use $K_i = \mathcal{O}(K/n)$ as an upper bound. ■

## E.4 Privacy-Utility Trade-offs of Private ADMM Algorithms

Now, we amalgamate the privacy analysis of the three private ADMM algorithms (Appendix E.3) with the generic convergence analysis of fixed-point iterations (Theorem 8.5) to obtain the privacy-utility trade-off for these three algorithms.

### E.4.1 Centralized Private ADMM

Here, we present the detailed proof of Corollary 8.8.

**Corollary 8.8.** *Under the assumptions and notations of Theorem 8.5 and 8.7, and for number of iterations* $K = \mathcal{O}\left(\log\left(\frac{L\gamma}{nD(1-\tau)}\left(\frac{p\alpha}{\varepsilon}\right)^{1/2} + \frac{p\alpha L^2\gamma^2}{\varepsilon n^2 D(1-\tau)^3}\right)\right)$*, Algorithm 8.2 achieves*

$$\mathbb{E}\left(\|u_{K+1} - u^*\|^2\right) = \widetilde{\mathcal{O}}\left(\frac{L\gamma\sqrt{p\alpha}}{\sqrt{\varepsilon}n\left(1-\tau\right)} + \frac{p\alpha L^2\gamma^2}{\varepsilon n^2\left(1-\tau\right)^3}\right). \tag{E.19}$$

*Proof.* We recall from Theorem 8.5 that

$$\begin{aligned}\mathbb{E}[\|u_{k+1} - u^*\|^2] &\leqslant \left(1 - \frac{q^2(1-\tau)}{8}\right)^k \left(D + \frac{2(\sigma\sqrt{p}+\zeta)}{\sqrt{q}\left(1-\tau\right)}\right) + \frac{8(p\sigma^2+\zeta^2)}{q^3(1-\tau)^3} + \frac{4}{(1-\tau)} \\ &\leq \left(1 - \frac{q^2(1-\tau)}{8}\right)^k D + \left(\frac{8(\sigma\sqrt{p}+\zeta)}{\sqrt{q}\left(1-\tau\right)} + \frac{8(p\sigma^2+\zeta^2)}{q^3(1-\tau)^3}\right)\end{aligned}$$

In case of centralized private ADMM, $\zeta = 0$, $q = 1$, and $\sigma^2 = \frac{8K\alpha L^2\gamma^2}{\varepsilon n^2}$ (Theorem 8.7). Thus, we obtain for $k = K$ that

$$\begin{aligned}\mathbb{E}\left(\|u_{K+1} - u^*\|^2\right) &\leq \left(\frac{7+\tau}{8}\right)^K D + \left(\frac{8\sqrt{p}}{(1-\tau)}\sqrt{\frac{8K\alpha L^2\gamma^2}{\varepsilon n^2}} + \frac{8p}{(1-\tau)^3}\left(\frac{8K\alpha L^2\gamma^2}{\varepsilon n^2}\right)\right) \\ &\leq \left(\frac{7+\tau}{8}\right)^K D + 2\left(\frac{4\sqrt{p}}{(1-\tau)}\sqrt{\frac{8\alpha L^2\gamma^2}{\varepsilon n^2}} + \frac{8p}{(1-\tau)^3}\left(\frac{8\alpha L^2\gamma^2}{\varepsilon n^2}\right)\right)K \\ &= \left(\frac{7+\tau}{8}\right)^K D + \mathcal{O}\left(\frac{L\gamma}{(1-\tau)\,n}\left(\frac{p\alpha}{\varepsilon}\right)^{1/2} + \frac{p\alpha L^2\gamma^2}{\varepsilon n^2\left(1-\tau\right)^3}\right)K\end{aligned}$$

Now, if we consider $K$ such that

$$\begin{aligned}\left(\frac{7+\tau}{8}\right)^K D &= \mathcal{O}\left(\frac{L\gamma}{(1-\tau)\,n}\left(\frac{p\alpha}{\varepsilon}\right)^{1/2} + \frac{p\alpha L^2\gamma^2}{\varepsilon n^2\left(1-\tau\right)^3}\right) \\ \implies K &= \mathcal{O}\left(\log\left(\frac{L\gamma}{nD\left(1-\tau\right)}\left(\frac{p\alpha}{\varepsilon}\right)^{1/2} + \frac{p\alpha L^2\gamma^2}{\varepsilon n^2 D\left(1-\tau\right)^3}\right)\right),\end{aligned}$$

we obtain

$$\begin{aligned}
&\mathbb{E}\left(\|u_{K+1}-u^*\|^2\right)\\
=&\mathcal{O}\left(\left(\frac{L\gamma}{n\left(1-\tau\right)}\left(\frac{p\alpha}{\varepsilon}\right)^{1/2}+\frac{p\alpha L^2\gamma^2}{\varepsilon n^2\left(1-\tau\right)^3}\right)\log\left(\frac{L\gamma}{nD\left(1-\tau\right)}\left(\frac{p\alpha}{\varepsilon}\right)^{1/2}+\frac{p\alpha L^2\gamma^2}{\varepsilon n^2 D\left(1-\tau\right)^3}\right)\right)\\
=&\widetilde{\mathcal{O}}\left(\frac{L\gamma\sqrt{p\alpha}}{\sqrt{\varepsilon}n\left(1-\tau\right)}+\frac{p\alpha L^2\gamma^2}{\varepsilon n^2\left(1-\tau\right)^3}\right)
\end{aligned}$$

■

### E.4.2 Federated Private ADMM with Subsampling

Here, we present the detailed proof of Corollary 8.11.

**Corollary 8.11.** *Under the assumptions and notations of Theorem 8.5 and 8.9, for number of iterations* $K=\mathcal{O}\left(\log\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon}rnD(1-\tau)}+\frac{p\alpha L^2\gamma^2}{\varepsilon r^2n^2D(1-\tau)^3}\right)\right)$*, and* $m=rn$ *for* $r\in(0,1)$*, Algorithm 8.3 achieves*

$$\mathbb{E}\left(\|u_{K+1}-u^*\|^2\right)=\widetilde{\mathcal{O}}\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon}rn\left(1-\tau\right)}+\frac{p\alpha L^2\gamma^2}{\varepsilon r^2n^2\left(1-\tau\right)^3}\right). \tag{E.20}$$

*Proof.* In case of federated private ADMM, $\zeta=0$, $q=\frac{m}{n}$, and $\sigma^2=\frac{16K\alpha L^2\gamma^2}{\varepsilon n^2}$ (Theorem 8.9). Thus, using Theorem 8.5, we obtain for $k=K$ that

$$\begin{aligned}
\mathbb{E}\left(\|u_{K+1}-u^*\|^2\right)&\leq\left(1-\frac{m^2(1-\tau)}{8n^2}\right)^K D+\left(\frac{8\sqrt{p}}{(1-\tau)}\sqrt{\frac{n}{m}}\sqrt{\frac{16K\alpha L^2\gamma^2}{\varepsilon n^2}}+\frac{8p}{(1-\tau)^3}\left(\frac{16K\alpha L^2\gamma^2}{\varepsilon n^2}\right)\left(\frac{n}{m}\right)^3\right)\\
&\leq\left(1-\frac{m^2(1-\tau)}{8n^2}\right)^K D+2\left(\frac{8\sqrt{p}}{(1-\tau)}\sqrt{\frac{8\alpha L^2\gamma^2}{\varepsilon nm}}+\frac{8p}{(1-\tau)^3}\left(\frac{8\alpha L^2\gamma^2}{\varepsilon n^2}\right)\frac{n}{m^3}\right)K\\
&=\left(1-\frac{m^2(1-\tau)}{8n^2}\right)^K D+\mathcal{O}\left(\frac{L\gamma}{(1-\tau)}\left(\frac{p\alpha}{\varepsilon nm}\right)^{1/2}+\frac{L^2\gamma^2p\alpha}{\varepsilon\left(1-\tau\right)^3}\frac{n}{m^3}\right)K\\
&=\left(1-\frac{m^2(1-\tau)}{8n^2}\right)^K D+\mathcal{O}\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon}rn\left(1-\tau\right)}+\frac{p\alpha L^2\gamma^2}{\varepsilon r^2n^2\left(1-\tau\right)^3}\right)K
\end{aligned}$$

The last equality holds true when we choose $m=rn$, where $r\in(0,1/5]$ is a constant subsampling ratio.

Now, if we consider $K=\mathcal{O}\left(\log\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon}rnD(1-\tau)}+\frac{p\alpha L^2\gamma^2}{\varepsilon r^2n^2D(1-\tau)^3}\right)\right)$, we obtain

$$\begin{aligned}
&\mathbb{E}\left(\|u_{K+1}-u^*\|^2\right)\\
=&\mathcal{O}\left(\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon}rn\left(1-\tau\right)}+\frac{p\alpha L^2\gamma^2}{\varepsilon r^2n^2\left(1-\tau\right)^3}\right)\log\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon}rnD\left(1-\tau\right)}+\frac{p\alpha L^2\gamma^2}{\varepsilon r^2n^2D\left(1-\tau\right)^3}\right)\right)
\end{aligned}$$

$$=\widetilde{\mathcal{O}}\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon r}n\left(1-\tau\right)}+\frac{p\alpha L^2\gamma^2}{\varepsilon r^2 n^2\left(1-\tau\right)^3}\right)$$

■

### E.4.3 Fully Decentralized Private ADMM

Here, we present the detailed proof of Corollary 8.13.

**Corollary 8.13.** *Under the assumptions and notations of Theorem 8.5 and 8.12, and for number of iterations* $K=\mathcal{O}\left(\log\left(\frac{L\gamma}{D(1-\tau)}\left(\frac{p\alpha\log n}{\varepsilon n}\right)^{1/2}+\frac{L^2\gamma^2}{D(1-\tau)^3}\left(\frac{p\alpha\log n}{\varepsilon n}\right)\right)\right)$*, Algorithm 8.4 achieves*

$$\mathbb{E}\left(\|u_{K+1}-u^*\|^2\right)=\widetilde{\mathcal{O}}\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon n}\left(1-\tau\right)}+\frac{p\alpha L^2\gamma^2}{\varepsilon n\left(1-\tau\right)^3}\right). \tag{E.21}$$

*Proof.* In case of decentralized private ADMM, $\zeta=0$, $q=\frac{1}{n}$, and $\sigma^2=\frac{8K_i\alpha L^2\gamma^2\log n}{\sigma^2 n}=\frac{8K\alpha L^2\gamma^2\log n}{\sigma^2 n^2}$ (Theorem 8.9). Thus, using Theorem 8.5, we obtain for $k=K$ that

$$\begin{aligned}
\mathbb{E}\left(\|u_{K+1}-u^*\|^2\right)&\leq\left(1-\frac{q^2(1-\tau)}{8}\right)^k D+\left(\frac{8(\sigma\sqrt{p}+\zeta)}{\sqrt{q}\left(1-\tau\right)}+\frac{8(p\sigma^2+\zeta^2)}{q^3(1-\tau)^3}\right)\\
&\leq\left(1-\frac{1-\tau}{8n^2}\right)^K D+\left(\frac{8\sqrt{p}}{(1-\tau)}\sqrt{n}\sqrt{\frac{8K\alpha L^2\gamma^2\log n}{\varepsilon n^2}}+\frac{8p}{(1-\tau)^3}\left(\frac{8K\alpha L^2\gamma^2\log n}{\varepsilon n^2}\right)n^3\right)\\
&\leq\left(1-\frac{1-\tau}{8n^2}\right)^K D+2\left(\frac{8\sqrt{p}}{(1-\tau)}\sqrt{\frac{8\alpha L^2\gamma^2\log n}{\varepsilon n}}+\frac{8p}{(1-\tau)^3}\left(\frac{8\alpha L^2\gamma^2\log n}{\varepsilon n}\right)\right)K\\
&=\left(1-\frac{1-\tau}{8n^2}\right)^K D+\mathcal{O}\left(\frac{L\gamma}{(1-\tau)}\left(\frac{p\alpha\log n}{\varepsilon n}\right)^{1/2}+\frac{L^2\gamma^2}{(1-\tau)^3}\left(\frac{p\alpha\log n}{\varepsilon n}\right)\right)K
\end{aligned}$$

Now, if we consider $K=\mathcal{O}\left(\log\left(\frac{L\gamma}{D(1-\tau)}\left(\frac{p\alpha\log n}{\varepsilon n}\right)^{1/2}+\frac{L^2\gamma^2}{D(1-\tau)^3}\left(\frac{p\alpha\log n}{\varepsilon n}\right)\right)\right)$, we obtain

$$\begin{aligned}
&\mathbb{E}\left(\|u_{K+1}-u^*\|^2\right)\\
=&\mathcal{O}\left(\left(\frac{L\gamma}{(1-\tau)}\sqrt{\frac{p\alpha\log n}{\varepsilon n}}+\frac{L^2\gamma^2}{(1-\tau)^3}\left(\frac{p\alpha\log n}{\varepsilon n}\right)\right)\log\left(\frac{L\gamma}{D\left(1-\tau\right)}\sqrt{\frac{p\alpha\log n}{\varepsilon n}}+\frac{L^2\gamma^2}{D\left(1-\tau\right)^3}\left(\frac{p\alpha\log n}{\varepsilon n}\right)\right)\right)\\
=&\widetilde{\mathcal{O}}\left(\frac{\sqrt{p\alpha}L\gamma}{\sqrt{\varepsilon n}\left(1-\tau\right)}+\frac{p\alpha L^2\gamma^2}{\varepsilon n\left(1-\tau\right)^3}\right)
\end{aligned}$$

■

# Bibliography


[Aba+16] Martin Abadi et al. "Deep Learning with Differential Privacy". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS '16. New York, NY, USA: Association for Computing Machinery, Oct. 2016, pp. 308–318. ISBN: 978-1-4503-4139-4. DOI: 10.1145/2976749.2978318. URL: https://doi.org/10.1145/2976749.2978318 (visited on 07/22/2021).

[Aim20] Olivier Aim. In: *Les théories de la surveillance*. Armand Colin, Nov. 2020, pp. 237–243.

[AJM19] Kareem Amin, Matthew Joseph, and Jieming Mao. "Pan-Private Uniformity Testing". In: *CoRR* abs/1911.01452 (2019). arXiv: 1911.01452. URL: http://arxiv.org/abs/1911.01452.

[All+24] Youssef Allouah et al. "The Privacy Power of Correlated Noise in Decentralized Learning". In: *ICML* (2024).

[AR20] Ghadir Ayache and Salim El Rouayheb. *Private Weighted Random Walk Stochastic Gradient Descent*. Tech. rep. arXiv:2009.01790, 2020.

[AT22] Jason Altschuler and Kunal Talwar. "Privacy of Noisy Stochastic Gradient Descent: More Iterations without More Privacy Loss". In: *NeurIPS*. 2022.

[At24] AlphaProof and AlphaGeometry teams. *AI solves IMO problems at silver medal level*. 2024. URL: https://deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level/.

[Bac24] Francis Bach. *Learning theory from first principles*. en. London, England: MIT Press, Dec. 2024.

[Bal+19a] Borja Balle et al. *Differentially Private Summation with Multi-Message Shuffling*. Tech. rep. arxiv:1906.09116, 2019.

[Bal+19b] Borja Balle et al. "The Privacy Blanket of the Shuffle Model". In: *CRYPTO*. 2019.

[Bal+20a] Victor Balcer et al. *Connecting Robust Shuffle Privacy and Pan-Privacy*. 2020. arXiv: 2004.09481 [cs.CR].

[Bal+20b] Borja Balle et al. "Privacy Amplification via Random Check-Ins". In: *NeurIPS*. 2020.

[BB19] Didier Bigo and Laurent Bonelli. ""We aren't a Big Brother!" The Authority and Legitimization Strategies of Intelligence Services in the Capture and Use of Digital Data: Autorité et stratégies de légitimation des services de renseignement dans la captation et l'usage des données numériques". In: *Cultures and conflits* 114–115 (Dec. 2019), 199–226. ISSN: 1777-5345. DOI: 10.4000/conflits.21180. URL: http://dx.doi.org/10.4000/conflits.21180.

[BBG18] Borja Balle, Gilles Barthe, and Marco Gaboardi. "Privacy Amplification by Subsampling: Tight Analyses via Couplings and Divergences". In: *NeurIPS*. 2018.

[BBG20] Raphaël Berthier, Francis Bach, and Pierre Gaillard. "Accelerated Gossip in Networks of Given Dimension Using Jacobi Polynomial Iterations". In: *SIAM Journal on Mathematics of Data Science* 2.1 (2020), pp. 24–47. DOI: 10.1137/19M1244822. eprint: https://doi.org/10.1137/19M1244822. URL: https://doi.org/10.1137/19M1244822.

[BC11] Heinz H. Bauschke and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2011.

[BCH22] Borja Balle, Giovanni Cherubin, and Jamie Hayes. "Reconstructing Training Data with Informed Adversaries". In: *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, May 2022. DOI: 10.1109/sp46214.2022.9833677. URL: http://dx.doi.org/10.1109/SP46214.2022.9833677.

[Bel+18] Aurélien Bellet et al. "Personalized and Private Peer-to-Peer Machine Learning". In: *AISTATS*. 2018.

[Bel+20] James Henry Bell et al. "Secure Single-Server Aggregation with (Poly)Logarithmic Overheads". In: *CCS*. 2020.

[BGN17] S. Benthall, S. Gürses, and H. Nissenbaum. *Contextual Integrity Through the Lens of Computer Science*. Foundations and Trends® in Privacy and Security Series. Now Publishers, 2017. ISBN: 9781680833843. URL: https://books.google.fr/books?id=CaNzswEACAAJ.

[Bis23] Sayan Biswas. "Understanding and optimizing the trade-off between privacy and utility from a foundational perspective". PhD thesis. Institut Polytechnique de Paris, 2023.

[Bis+24] Sayan Biswas et al. "Beyond Noise: Privacy-Preserving Decentralized Learning with Virtual Nodes". In: *arXiv preprint arXiv:2404.09536* (2024).

[BM24] Simone Bombari and Marco Mondelli. *Privacy for Free in the Over-Parameterized Regime*. 2024. arXiv: 2410.14787 [stat.ML]. URL: https://arxiv.org/abs/2410.14787.

[BN15] Finn Brunton and Helen Nissenbaum. *Obfuscation : a user's guide for privacy and protest*. eng. Cambridge, Massachusetts: The MIT Press, 2015 - 2015. ISBN: 9780262029735.

[Boe05] Pieter Boeder. "Habermas' heritage: The future of the public sphere in the network society". In: (2005). URL: https://firstmonday.org/ojs/index.php/fm/article/download/1280/1200.

[Boe+21] Franziska Boenisch et al. "When the Curious Abandon Honesty: Federated Learning Is Not Private". In: *ArXiv* abs/2112.02918 (2021).

[Bon+17] Keith Bonawitz et al. "Practical Secure Aggregation for Privacy-Preserving Machine Learning". In: *CCS*. 2017.

[Bon+19] Kallista A. Bonawitz et al. "Towards Federated Learning at Scale: System Design". In: *CoRR* abs/1902.01046 (2019). arXiv: 1902.01046. URL: http://arxiv.org/abs/1902.01046.

[Boy+06] Stephen Boyd et al. "Randomized gossip algorithms". In: *IEEE transactions on information theory* 52.6 (2006), pp. 2508–2530.

[Boy+11] Stephen Boyd et al. "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers". In: *Foundations and Trends® in Machine Learning* 3.1 (2011), pp. 1–122.

[BP16] Albert-László Barabási and Márton Pósfai. *Network science*. Cambridge: Cambridge University Press, 2016. ISBN: 9781107076266 1107076269. URL: http://barabasi.com/networksciencebook/.

[BP86] Ronald L. Breiger and Philippa E. Pattison. "Cumulated social roles: The duality of persons and their algebras". In: *Social Networks* 8.3 (1986), pp. 215–256.

[BST14] Raef Bassily, Adam Smith, and Abhradeep Thakurta. "Private Empirical Risk Minimization: Efficient Algorithms and Tight Error Bounds". In: *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. Philadelphia, PA, USA: IEEE, Oct. 2014, pp. 464–473. ISBN: 978-1-4799-6517-5. DOI: 10.1109/FOCS.2014.56. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6979031 (visited on 11/06/2019).

[Bur16] Jenna Burrell. "How the machine 'thinks': Understanding opacity in machine learning algorithms". In: *Big Data and Society* 3.1 (Jan. 2016), p. 205395171562251. ISSN: 2053-9517. DOI: 10.1177/2053951715622512. URL: http://dx.doi.org/10.1177/2053951715622512.

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[Byr03] Charles Byrne. "A unified treatment of some iterative algorithms in signal processing and image reconstruction". In: *Inverse Problems* 20.1 (2003), pp. 103–120.

[BZ06] MICHAEL BARBARO and TOM ZELLER. *A Face Is Exposed for AOL Searcher No. 4417749*. 2006. URL: http://www.nytimes.com/2006/08/09/technology/09aol.html.

[Cao+21] Xuanyu Cao et al. "Differentially Private ADMM for Regularized Consensus Optimization". In: *IEEE Transactions on Automatic Control (TAC)* 66.8 (2021), pp. 3718–3725.

[Car+21] Nicholas Carlini et al. "Membership Inference Attacks From First Principles". In: *2022 IEEE Symposium on Security and Privacy (SP)* (2021), pp. 1897–1914. URL: https://api.semanticscholar.org/CorpusID:244920593.

[Cau+47] Augustin Cauchy et al. "Méthode générale pour la résolution des systemes d'équations simultanées". In: *Comp. Rend. Sci. Paris* 25.1847 (1847), pp. 536–538.

[CB22] Edwige Cyffers and Aurélien Bellet. "Privacy amplification by decentralization". In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR. 2022, pp. 5334–5353.

[CBB23] Edwige Cyffers, Aurélien Bellet, and Debabrota Basu. "From noisy fixed-point iterations to private ADMM for centralized and federated learning". In: *International Conference on Machine Learning (ICML)* (2023), pp. 6683–6711.

[CBP24] Tudor Cebere, Aurélien Bellet, and Nicolas Papernot. "Tighter Privacy Auditing of DP-SGD in the Hidden State Threat Model". In: *ArXiv* abs/2405.14457 (2024). URL: https://api.semanticscholar.org/CorpusID:269982111.

[CBU24] Edwige Cyffers, Aurélien Bellet, and Jalaj Upadhyay. "Differentially Private Decentralized Learning with Random Walks". In: *International Conference on Machine Learning (ICML)* (2024).

[Cha+13] Konstantinos Chatzikokolakis et al. "Broadening the Scope of Differential Privacy Using Metrics". In: *The 13th Privacy Enhancing Technologies Symposium*. Ed. by De Cristofaro et al. Vol. 7981. Lecture Notes in Computer Science. Bloomington, Indiana, United States: Springer, July 2013, pp. 82–102. DOI: 10.1007/978-3-642-39077-7. URL: https://hal.inria.fr/hal-00767210.

[Che+19a] Hsin-Pai Cheng et al. "Towards Decentralized Deep Learning with Differential Privacy". In: *CLOUD*. 2019.

[Che+19b] Albert Cheu et al. "Distributed Differential Privacy via Shuffling". In: *EUROCRYPT*. 2019.

[Che+21] Bo Chen et al. "Edge Differential Privacy for Algebraic Connectivity of Graphs". In: *2021 60th IEEE Conference on Decision and Control (CDC)*. Austin, TX, USA: IEEE Press, 2021, 2764–2769. DOI: 10.1109/CDC45484.2021.9683306. URL: https://doi.org/10.1109/CDC45484.2021.9683306.

[CMS11] Kamalika Chaudhuri, Claire Monteleoni, and Anand D. Sarwate. "Differentially Private Empirical Risk Minimization". In: *Journal of Machine Learning Research* 12.29 (2011), pp. 1069–1109. ISSN: 1533-7928. URL: http://jmlr.org/papers/v12/chaudhuri11a.html (visited on 06/29/2021).

[CNI15] CNIL. *La loi Informatique et Libertés*. Accessed: 2024-09-26. 2015. URL: https://www.cnil.fr/fr/la-loi-informatique-et-libertes.

[Con01] Congress. *UNITING AND STRENGTHENING AMERICA BY PROVIDING APPROPRIATE TOOLS REQUIRED TO INTERCEPT AND OBSTRUCT TERRORISM (USA PATRIOT ACT)*. 2001. URL: https://www.congress.gov/107/plaws/publ56/PLAW-107publ56.htm.

[CP19] Patrick L Combettes and Jean-Christophe Pesquet. "Stochastic quasi-Fejér block-coordinate fixed point iterations with random sweeping II: mean-square and linear convergence". In: *Mathematical Programming* 174.1 (2019), pp. 433–451.

[CP21] Patrick L. Combettes and Jean-Christophe Pesquet. "Fixed Point Strategies in Data Science". In: *IEEE Transactions on Signal Processing* 69 (2021), 3878–3905. ISSN: 1941-0476. DOI: 10.1109/tsp.2021.3069677. URL: http://dx.doi.org/10.1109/TSP.2021.3069677.

[CSS12a] T.-H. Hubert Chan, Elaine Shi, and Dawn Song. "Optimal Lower Bound for Differentially Private Multi-party Aggregation". In: *ESA*. 2012.

[CSS12b] T.-H. Hubert Chan, Elaine Shi, and Dawn Song. "Privacy-Preserving Stream Aggregation with Fault Tolerance". In: *Financial Cryptography*. 2012.

[Cyf+22] Edwige Cyffers et al. "Muffliato: Peer-to-peer privacy amplification for decentralized optimization and averaging". In: *Advances in Neural Information Processing Systems (NeurIPS)* 35 (2022), pp. 15889–15902.

[Cyf+24] Edwige Cyffers et al. "Optimal Classification under Performative Distribution Shift". In: *Advances in Neural Information Processing Systems (NeurIPS)* 37 (2024).

[Dan+22] Yatin Dandi et al. *Data-heterogeneity-aware Mixing for Decentralized Learning*. arXiv:2204.06477. 2022.

[DEC23] Florine W. Dekker, Zekeriya Erkin, and Mauro Conti. "Topology-Based Reconstruction Prevention for Decentralised Learning". In: *ArXiv* abs/2312.05248 (2023). URL: https://api.semanticscholar.org/CorpusID:266149467.

[Des14] Alain Desrosières. *Prouver et gouverner: Une analyse politique des statistiques publiques*. La Découverte, Apr. 2014. ISBN: 9782707178954. DOI: 10.3917/dec.desro.2014.01. URL: http://dx.doi.org/10.3917/dec.desro.2014.01.

[Des88] Alain Desrosières. "Masses, individus, moyennes : la statistique sociale au XIXe siècle". In: *Hermès* n° 2.2 (1988), p. 41. ISSN: 1963-1006. DOI: 10.4267/2042/15681. URL: http://dx.doi.org/10.4267/2042/15681.

[Dim+10] A. G. Dimakis et al. "Gossip Algorithms for Distributed Signal Processing". In: *Proceedings of the IEEE* 98.11 (2010), pp. 1847–1864.

[Din+20] Jiahao Ding et al. "Towards Plausible Differentially Private ADMM Based Distributed Machine Learning". In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (2020).

[DJW13] John C Duchi, Michael I Jordan, and Martin J Wainwright. "Local privacy and statistical minimax rates". In: *FOCS*. 2013.

[DMS04] Roger Dingledine, Nick Mathewson, and Paul Syverson. "Tor: the second-generation onion router". In: *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*. 2004.

[DR14] Cynthia Dwork and Aaron Roth. "The Algorithmic Foundations of Differential Privacy". In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (Aug. 2014), pp. 211–407. ISSN: 1551-305X. DOI: 10.1561/0400000042. URL: https://doi.org/10.1561/0400000042 (visited on 05/16/2023).

[DRV10] Cynthia Dwork, Guy N. Rothblum, and Salil Vadhan. "Boosting and Differential Privacy". In: *FOCS*. 2010.

[Dwo+06a] Cynthia Dwork et al. "Calibrating Noise to Sensitivity in Private Data Analysis". In: *Theory of Cryptography*. Ed. by Shai Halevi and Tal Rabin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 265–284. ISBN: 978-3-540-32732-5.

[Dwo+06b] Cynthia Dwork et al. "Our Data, Ourselves: Privacy Via Distributed Noise Generation". In: *EUROCRYPT*. 2006.

[Dwo+10] Cynthia Dwork et al. "Pan-Private Streaming Algorithms". In: *ICS*. 2010.

[EH08] Ernesto Estrada and Naomichi Hatano. "Communicability in complex networks". In: *Physical Review E* 77.3 (2008). DOI: 10.1103/physreve.77.036111. URL: https://doi.org/10.1103\%2Fphysreve.77.036111.

[EHM20] Mathieu Even, Hadrien Hendrikx, and Laurent Massoulié. *Asynchrony and Acceleration in Gossip Algorithms*. Tech. rep. arXiv:2011.02379, 2020.

[EK15] Ernesto Estrada and Philip A. Knight. *A First Course in Network Theory*. English. United Kingdom: Oxford University Press, Mar. 2015. ISBN: 978-0-19-872645-6.

[EM22] Ksenia Ermoshina and Francesca Musiani. "Safer spaces by design? Federated architectures and alternative socio-technical models for content moderation". In: *Annual Symposium of the Global Internet Governance Academic Network (GigaNet)*. Addis-Ababa, Ethiopia, Nov. 2022. URL: https://hal.science/hal-03930548.

[EMS22] Mathieu Even, Laurent Massoulié, and Kevin Scaman. "On Sample Optimality in Personalized Collaborative and Federated Learning". In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh et al. 2022. URL: https://openreview.net/forum?id=7EP90NMAoK.

[ER59] P. Erdös and A. Rényi. "On Random Graphs I". In: *Publicationes Mathematicae Debrecen* 6 (1959), p. 290.

[Erl+19] Ulfar Erlingsson et al. "Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity". In: *SODA*. 2019.

[Eve+21] Mathieu Even et al. "Continuized Accelerations of Deterministic and Stochastic Gradient Descents, and of Gossip Algorithms". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 28054–28066. URL: https://proceedings.neurips.cc/paper/2021/file/ec26fc2eb2b75aece19c70392dc744c2-Paper.pdf.

[Eve23] Mathieu Even. *Stochastic Gradient Descent under Markovian Sampling Schemes*. arXiv:2302.14428 [cs, math]. Feb. 2023. DOI: 10.48550/arXiv.2302.14428. URL: http://arxiv.org/abs/2302.14428 (visited on 03/28/2023).

[EY15] Jonathan Eckstein and Wang Yao. *Understanding the Convergence of the Alternating Direction Method of Multipliers: Theoretical and Computational Perspectives*. 2015.

[Fel+18] Vitaly Feldman et al. "Privacy Amplification by Iteration". In: *FOCS*. 2018.

[FMT20] Vitaly Feldman, Audra McMillan, and Kunal Talwar. *Hiding Among the Clones: A Simple and Nearly Optimal Analysis of Privacy Amplification by Shuffling*. Tech. rep. arXiv:2012.12803, 2020.

[Fra+21] Yann Fraboni et al. "Clustered Sampling: Low-Variance and Improved Representativity for Clients Selection in Federated Learning". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 3407–3416. URL: https://proceedings.mlr.press/v139/fraboni21a.html.

[Ful] *FullStory : un outil d'analyse du parcours des utilisateurs*. 2020. URL: https://junto.fr/blog/fullstory/.

[FZ20] Vitaly Feldman and Tijana Zrnic. "Individual Privacy Accounting via a Renyi Filter". In: *Neural Information Processing Systems*. 2020. URL: https://api.semanticscholar.org/CorpusID:221293138.

[FZ21] Vitaly Feldman and Tijana Zrnic. "Individual Privacy Accounting via a Rényi Filter". In: *NeurIPS*. 2021.

[GB14] Pontus Giselsson and Stephen P. Boyd. "Diagonal scaling in Douglas-Rachford splitting and ADMM". In: *CDC*. 2014.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[Gei+20a] Jonas Geiping et al. "Inverting Gradients - How Easy is It to Break Privacy in Federated Learning?" In: *NeurIPS*. 2020.

[Gei+20b] Jonas Geiping et al. "Inverting Gradients - How easy is it to break privacy in federated learning?" In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 16937–16947. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/c4ede56bbd98819ae6112b20ac6bf145-Paper.pdf.

[GFB16] Pontus Giselsson, Mattias Fält, and Stephen P. Boyd. "Line search for averaged operator iteration". In: *2016 IEEE 55th Conference on Decision and Control (CDC)* (2016), pp. 1015–1022. URL: https://api.semanticscholar.org/CorpusID:15628151.

[GG23] Guillaume Garrigos and Robert M. Gower. *Handbook of Convergence Theorems for (Stochastic) Gradient Methods*. arXiv:2301.11235 [math]. Jan. 2023. DOI: 10.48550/arXiv.2301.11235. URL: http://arxiv.org/abs/2301.11235 (visited on 02/07/2023).

[Gha+20] Badih Ghazi et al. *Pure Differentially Private Summation from Anonymous Messages*. Tech. rep. arXiv:2002.01919, 2020.

[Gow+19] Robert Mansel Gower et al. "SGD: General Analysis and Improved Rates". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 5200–5209. URL: https://proceedings.mlr.press/v97/qian19b.html.

[GR01] Chris Godsil and Gordon F Royle. *Algebraic graph theory*. Vol. 207. Springer Science & Business Media, 2001.

[Gua24] The Guardian. "Google keeps location history data on abortion clinics despite delete pledge". In: *The Guardian* (2024). URL: https://www.theguardian.com/technology/2024/jan/16/google-keeps-location-history-data-abortion-clinics-despite-delete-pledge.

[Gue+23] Rachid Guerraoui et al. "On the Inherent Anonymity of Gossiping". In: *DISC* (2023).

[Had+21] Farzin Haddadpour et al. "Federated Learning with Compression: Unified Analysis and Sharp Guarantees". In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, 2021, pp. 2350–2358. URL: https://proceedings.mlr.press/v130/haddadpour21a.html.

[Hao20] Karen Hao. "Algorithms create a poverty trap. Lawyers fight back". In: *MIT Technology Review* (2020). URL: https://www.technologyreview.com/2020/12/04/1013068/algorithms-create-a-poverty-trap-lawyers-fight-back/.

[HBM20] Hadrien Hendrikx, Francis R. Bach, and Laurent Massouli'e. "Dual-Free Stochastic Decentralized Optimization with Variance Reduction". In: *ArXiv* abs/2006.14384 (2020). URL: https://api.semanticscholar.org/CorpusID:220055757.

[Hen22] Hadrien Hendrikx. *A principled framework for the design and analysis of token algorithms*. Number: arXiv:2205.15015 arXiv:2205.15015 [cs, math]. May 2022. DOI: 10.48550/arXiv.2205.15015. URL: http://arxiv.org/abs/2205.15015 (visited on 09/03/2022).

[HJ12] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *arXiv preprint arxiv:2006.11239* (2020).

[HKP19] Christopher Hoffman, Matthew Kahle, and Elliot Paquette. "Spectral Gaps of Random Graphs and Applications". In: *International Mathematics Research Notices* 2021.11 (2019), 8353–8404. ISSN: 1687-0247. DOI: 10.1093/imrn/rnz077. URL: http://dx.doi.org/10.1093/imrn/rnz077.

[HM16] Eszter Hargittai and Alice Marwick. "'"What Can I Really Do?" : Explaining the Privacy Paradox with Online Apathy". In: *International Journal of Communication* 10 (2016), pp. 3737–3757. ISSN: 1932-8036. URL: https://doi.org/10.5167/uzh-148157.

[HMV15] Zhenqi Huang, Sayan Mitra, and Nitin Vaidya. "Differentially Private Distributed Optimization". In: *ICDCN*. 2015.

[Hu+19] Yaochen Hu et al. "Learning privately over distributed features: An ADMM sharing approach". In: *arXiv preprint arXiv:1907.07735* (2019).

[Hua+19] Zonghao Huang et al. "DP-ADMM: ADMM-Based Distributed Learning With Differential Privacy". In: *IEEE Transactions on Information Forensics and Security* 15 (2019), pp. 1002–1012.

[Jay+18] Bargav Jayaraman et al. "Distributed Learning without Distress: Privacy-Preserving Empirical Risk Minimization". In: *NeurIPS*. 2018.

[Jhu+22] Divyansh Jhunjhunwala et al. "Fedvarp: Tackling the variance due to partial client participation in federated learning". In: *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*. Ed. by James Cussens and Kun Zhang. Vol. 180. Proceedings of Machine Learning Research. PMLR, 2022, pp. 906–916. URL: https://proceedings.mlr.press/v180/jhunjhunwala22a.html.

[JRJ10] Björn Johansson, Maben Rabi, and Mikael Johansson. "A Randomized Incremental Subgradient Method for Distributed Optimization in Networked Systems". In: *SIAM Journal on Optimization* 20.3 (2010), pp. 1157–1170. DOI: 10.1137/08073038X.

[KÁF23] Raouf Kerkouche, Gergely Ács, and Mario Fritz. "Client-specific Property Inference against Secure Aggregation in Federated Learning". In: *ArXiv* abs/2303.03908 (2023).

[Kai+21] Peter Kairouz et al. "Advances and Open Problems in Federated Learning". In: *Foundations and Trends® in Machine Learning* 14.1–2 (2021), pp. 1–210. ISSN: 1935-8237. DOI: 10.1561/2200000083. URL: http://dx.doi.org/10.1561/2200000083.

[Kar+20] Sai Praneeth Karimireddy et al. "SCAFFOLD: Stochastic Controlled Averaging for Federated Learning". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 5132–5143. URL: https://proceedings.mlr.press/v119/karimireddy20a.html.

[Kar+23] Sanjay Kariyappa et al. "Cocktail Party Attack: Breaking Aggregation-Based Privacy in Federated Learning using Independent Component Analysis". In: *ICML*. 2023.

[Kas+08] Shiva Prasad Kasiviswanathan et al. "What Can We Learn Privately?" In: *FOCS*. 2008.

[Kat53] Leo Katz. "A new status index derived from sociometric analysis". In: *Psychometrika* 18 (1953), pp. 39–43.

[Kol+20] Anastasia Koloskova et al. "A Unified Theory of Decentralized SGD with Changing Topology and Local Updates". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 5381–5393. URL: https://proceedings.mlr.press/v119/koloskova20a.html.

[Kon+21] Lingjing Kong et al. "Consensus Control for Decentralized Deep Learning". In: *ICML*. 2021.

[KOV14] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. "Extremal mechanisms for local differential privacy". In: *NIPS*. 2014.

[Kri09] Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. 2009. URL: https://api.semanticscholar.org/CorpusID:18268744.

[KSJ19] Anastasia Koloskova, Sebastian U. Stich, and Martin Jaggi. "Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication". In: *ICML*. 2019.

[KTH22] Antti Koskela, Marlon Tobaben, and Antti Honkela. "Individual Privacy Accounting with Gaussian Differential Privacy". In: *ArXiv* abs/2209.15596 (2022). URL: https://api.semanticscholar.org/CorpusID:252668508.

[LB+23] Batiste Le Bars et al. "Refined Convergence and Topology Learning for Decentralized SGD with Heterogeneous Data". In: *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*. Ed. by Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent. Vol. 206. Proceedings of Machine Learning Research. PMLR, 2023, pp. 1672–1702. URL: https://proceedings.mlr.press/v206/le-bars23a.html.

[Les06] Lawrence Lessig. *Code: Version 2.0*. eng. New York: Basic Books, 2006. ISBN: 978-0-465-03914-2. URL: http://codev2.cc/.

[LFP15] Jingwei Liang, Jalal Fadili, and Gabriel Peyré. "Convergence rates with inexact non-expansive operators". In: *Mathematical Programming* 159.1-2 (2015), 403–434.

[Li+18] Chencheng Li et al. "Differentially Private Distributed Online Learning". In: *IEEE Transactions on Knowledge and Data Engineering* (2018).

[Li+20] Tian Li et al. "Federated Learning: Challenges, Methods, and Future Directions". In: *IEEE Signal Processing Magazine* 37.3 (May 2020), 50–60. ISSN: 1558-0792. DOI: 10.1109/msp.2020.2975749. URL: http://dx.doi.org/10.1109/MSP.2020.2975749.

[Lia+17] Xiangru Lian et al. "Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, 5336–5346. ISBN: 9781510860964.

[Lie+22] Seng Pei Liew et al. "Network shuffling: Privacy amplification via random walks". In: *Proceedings of the 2022 International Conference on Management of Data*. 2022, pp. 773–787.

[Lin24] Sander van der Linden. *Foolproof*. en. New York, NY: WW Norton, Feb. 2024.

[LK14] Jure Leskovec and Andrej Krevl. *SNAP Datasets: Stanford Large Network Dataset Collection*. http://snap.stanford.edu/data. June 2014.

[LLV07] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. "t-Closeness: Privacy Beyond k-Anonymity and l-Diversity". In: *2007 IEEE 23rd International Conference on Data Engineering*. IEEE, Apr. 2007. DOI: 10.1109/icde.2007.367856. URL: http://dx.doi.org/10.1109/ICDE.2007.367856.

[LM12] Jure Leskovec and Julian Mcauley. "Learning to Discover Social Circles in Ego Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/file/7a614fd06c325499f1680b9896beedeb-Paper.pdf.

[LM79] P. L. Lions and B. Mercier. "Splitting Algorithms for the Sum of Two Nonlinear Operators". In: *SIAM Journal on Numerical Analysis* 16.6 (1979), pp. 964–979.

[LP17] D.A. Levin and Y. Peres. *Markov Chains and Mixing Times*. MBK. American Mathematical Society, 2017. ISBN: 9781470429621. URL: https://books.google.fr/books?id=f208DwAAQBAJ.

[LS07] Cassio G. Lopes and Ali H. Sayed. "Incremental Adaptive Strategies Over Distributed Networks". In: *IEEE Transactions on Signal Processing* 55.8 (2007), pp. 4064–4077. DOI: 10.1109/TSP.2007.896034.

[M.16] Christopher M. *Pattern Recognition and Machine Learning*. en. Information Science and Statistics. New York, NY: Springer, Aug. 2016.

[Mac+07] Ashwin Machanavajjhala et al. "L-diversity: Privacy beyond k-anonymity". In: *ACM Transactions on Knowledge Discovery from Data* 1.1 (Mar. 2007), p. 3. ISSN: 1556-472X. DOI: 10.1145/1217299.1217302. URL: http://dx.doi.org/10.1145/1217299.1217302.

[MAK20] Sandra C Matz, Ruth E Appel, and Michal Kosinski. "Privacy in the age of psychological targeting". In: *Current Opinion in Psychology* 31 (2020). Privacy and Disclosure, Online and in Social Interactions, pp. 116–121. ISSN: 2352-250X. DOI: https://doi.org/10.1016/j.copsyc.2019.08.010. URL: https://www.sciencedirect.com/science/article/pii/S2352250X19301332.

[Man+22] Paul Mangold et al. "Differentially Private Coordinate Descent for Composite Empirical Risk Minimization". In: *ICML*. 2022.

[Mao+20] Xianghui Mao et al. "Walkman: A Communication-Efficient Random-Walk Algorithm for Decentralized Optimization". In: *IEEE Transactions on Signal Processing* 68 (2020), pp. 2513–2528.

[Mar+20] Othmane Marfoq et al. "Throughput-Optimal Topology Design for Cross-Silo Federated Learning". In: *NeurIPS*. 2020.

[MCB24] Abdellah El Mrini, Edwige Cyffers, and Aurélien Bellet. "Privacy Attacks in Decentralized Learning". In: *International Conference on Machine Learning (ICML)* (2024).

[McM+17] Brendan McMahan et al. "Communication-Efficient Learning of Deep Networks from Decentralized Data". In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, 2017, pp. 1273–1282. URL: https://proceedings.mlr.press/v54/mcmahan17a.html.

[McM+18] H. Brendan McMahan et al. "Learning Differentially Private Recurrent Language Models". In: *ICLR*. 2018.

[Mel+19] Luca Melis et al. "Exploiting Unintended Feature Leakage in Collaborative Learning". In: *2019 IEEE Symposium on Security and Privacy (SP)*. 2019, pp. 691–706. DOI: 10.1109/SP.2019.00029.

[Mir17] Ilya Mironov. "Rényi Differential Privacy". In: *2017 IEEE 30th Computer Security Foundations Symposium (CSF)* (Aug. 2017), pp. 263–275. DOI: 10.1109/CSF.2017.11. arXiv: 1702.07476. URL: http://arxiv.org/abs/1702.07476 (visited on 11/27/2020).

[Moh+91] Mohar et al. "The Laplacian spectrum of graphs". In: *Graph theory, combinatorics, and applications* (1991).

[Mon+13] Yves-Alexandre de Montjoye et al. "Unique in the Crowd: The privacy bounds of human mobility". In: *Scientific Reports* 3.1 (2013), p. 1376. ISSN: 2045-2322. DOI: 10.1038/srep01376. URL: https://doi.org/10.1038/srep01376.

[MR16] Aryan Mokhtari and Alejandro Ribeiro. "DSA: Decentralized Double Stochastic Averaging Gradient Algorithm". In: *Journal of Machine Learning Research* 17.61 (2016), pp. 1–35. URL: http://jmlr.org/papers/v17/15-292.html.

[MTZ19] Ilya Mironov, Kunal Talwar, and Li Zhang. "Rényi Differential Privacy of the Sampled Gaussian Mechanism". In: *arXiv:1908.10530 [cs, stat]* (Aug. 2019). arXiv: 1908.10530 [cs, stat]. URL: http://arxiv.org/abs/1908.10530 (visited on 11/21/2021).

[MV14] Aravindh Mahendran and Andrea Vedaldi. *Understanding Deep Image Representations by Inverting Them*. 2014. arXiv: 1412.0035 [cs.CV].

[Nak+19] Preetum Nakkiran et al. *Deep Double Descent: Where Bigger Models and More Data Hurt*. 2019. DOI: 10.48550/ARXIV.1912.02292. URL: https://arxiv.org/abs/1912.02292.

[Nas+21] Milad Nasr et al. "Adversary Instantiation: Lower Bounds for Differentially Private Machine Learning". In: *2021 IEEE Symposium on Security and Privacy (SP)* (2021), pp. 866–882. URL: https://api.semanticscholar.org/CorpusID:231583084.

[Nas+23] Milad Nasr et al. "Tight Auditing of Differentially Private Machine Learning". In: *32nd USENIX Security Symposium (USENIX Security 23)*. Anaheim, CA: USENIX Association, Aug. 2023, pp. 1631–1648. ISBN: 978-1-939133-37-3. URL: https://www.usenix.org/conference/usenixsecurity23/presentation/nasr.

[NBD22] Maxence Noble, Aurélien Bellet, and Aymeric Dieuleveut. "Differentially Private Federated Learning on Heterogeneous Data". In: *AISTATS*. 2022.

[NC15] Emmanuel Netter and Aurore Chaigneau. *Les biens numériques*. Ed. by Emmanuel Netter and Aurore Chaigneau. CEPRISCA, 2015. URL: https://hal.parisnanterre.fr/hal-01644198.

[ND88] Ben Noble and James W. Daniel. *Applied Linear Algebra*. 3rd. Pearson, 1988.

[Neg+19] Giovanni Neglia et al. "The Role of Network Topology for Distributed Machine Learning". In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. 2019, pp. 2350–2358. DOI: 10.1109/INFOCOM.2019.8737602.

[Neg+20] Giovanni Neglia et al. "Decentralized gradient methods: does topology matter?" In: *AISTATS*. 2020.

[Nes13] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Applied Optimization. Springer US, 2013. ISBN: 9781441988539. URL: https://books.google.fr/books?id=2-ElBQAAQBAJ.

[Ngu+23] Truc D. T. Nguyen et al. "Active Membership Inference Attack under Local Differential Privacy in Federated Learning". In: *ArXiv* abs/2302.12685 (2023). URL: https://api.semanticscholar.org/CorpusID:257205978.

[Nis09] Helen Nissenbaum. *Privacy in context*. Palo Alto, CA: Stanford University Press, Nov. 2009.

[NS06] Arvind Narayanan and Vitaly Shmatikov. "How To Break Anonymity of the Netflix Prize Dataset". In: *CoRR* abs/cs/0610105 (2006). arXiv: cs/0610105. URL: http://arxiv.org/abs/cs/0610105.

[NU48] Assemblée générale dees Nations Unies. *Déclaration Universelle des Droits de l'Homme*. 1948.

[Ohm09] Paul Ohm. "Broken Promises of Privacy: Responding to the Surprising Failure of Anonymization". In: (2009). URL: https://ssrn.com/abstract=1450006.

[OP16] Paul Ohm and Scott Peppet. "What If Everything Reveals Everything?" In: *Big Data Is Not a Monolith*. The MIT Press, Oct. 2016, 45–60. ISBN: 9780262335768. DOI: 10.7551/mitpress/10309.003.0010. URL: http://dx.doi.org/10.7551/mitpress/10309.003.0010.

[OT09] Alex Olshevsky and John N. Tsitsiklis. "Convergence Speed in Distributed Consensus and Averaging". In: *SIAM Journal on Control and Optimization* 48.1 (2009), pp. 33–55.

[Par12] E. Pariser. *The Filter Bubble: What the Internet is Hiding from You*. Penguin Books, 2012. ISBN: 9780241954522. URL: https://books.google.fr/books?id=Qn2ZnjzCE3gC.

[Pau+22] Matthias Paulik et al. *Federated Evaluation and Tuning for On-Device Personalization: System Design & Applications*. 2022. URL: https://arxiv.org/pdf/2102.08503.pdf.

[PRT23] Dario Pasquini, Mathilde Raynal, and Carmela Troncoso. "On the (In)security of Peer-to-Peer Decentralized Machine Learning". In: *IEEE Symposium on Security and Privacy (S&P)*. 2023.

[RA12] Arun Rajkumar and Shivani Agarwal. "A Differentially Private Stochastic Gradient Descent Algorithm for Multiparty Classification". In: *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Neil D. Lawrence and Mark Girolami. Vol. 22. Proceedings of Machine Learning Research. La Palma, Canary Islands: PMLR, 2012, pp. 933–941. URL: https://proceedings.mlr.press/v22/rajkumar12.html.

[Ram24] Ignacio Ramonet. *L'empire de la surveillance. Suivi de deux entretiens avec Julian Assange et Noam Chomsky*. Ed. by Gallimard. 2024.

[RD23] Beate Roessler and Judith DeCew. "Privacy". In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta and Uri Nodelman. Winter 2023. Metaphysics Research Lab, Stanford University, 2023.

[Red+20] Sashank J. Reddi et al. "Adaptive Federated Optimization". In: *CoRR* abs/2003.00295 (2020). arXiv: 2003.00295. URL: https://arxiv.org/abs/2003.00295.

[Rie+20] Nicola Rieke et al. "The future of digital health with federated learning". In: *npj Digital Medicine* 3.1 (Sept. 2020). ISSN: 2398-6352. DOI: 10.1038/s41746-020-00323-1. URL: http://dx.doi.org/10.1038/s41746-020-00323-1.

[RK22] Minseok Ryu and Kibaek Kim. "Differentially Private Federated Learning via Inexact ADMM with Multiple Local Updates". In: *arXiv e-prints*, arXiv:2202.09409 (2022). arXiv: 2202.09409 [cs.LG].

[RNV09] S. Sundhar Ram, A. Nedić, and V. V. Veeravalli. "Incremental stochastic subgradient algorithms for convex optimization". In: *SIAM Journal on Optimization* 20.2 (2009), pp. 691–717.

[Rob00] John M Roberts. "Correspondence analysis of two-mode network data". In: *Social Networks* 22.1 (2000), pp. 65–72. ISSN: 0378-8733. DOI: https://doi.org/10.1016/S0378-8733(00)00017-4. URL: https://www.sciencedirect.com/science/article/pii/S0378873300000174.

[Roc93] R. Tyrrell Rockafellar. "Lagrange Multipliers and Optimality". In: *SIAM Review* 35.2 (1993), pp. 183–238. DOI: 10.1137/1035044. eprint: https://doi.org/10.1137/1035044. URL: https://doi.org/10.1137/1035044.

[Rog+16] Ryan M. Rogers et al. "Privacy Odometers and Filters: Pay-as-you-Go Composition". In: *Neural Information Processing Systems*. 2016. URL: https://api.semanticscholar.org/CorpusID:1675786.

[Ryu+20] Ernest K. Ryu et al. "Operator Splitting Performance Estimation: Tight Contraction Factors and Optimal Parameter Selection". In: *SIAM Journal on Optimization* 30.3 (2020), pp. 2251–2271.

[SAW13] Latanya Sweeney, Akua Abu, and Julia Winn. "Identifying Participants in the Personal Genome Project by Name". In: *Innovation Law & Policy eJournal* (2013). URL: https://api.semanticscholar.org/CorpusID:316616.

[SBR22] César Sabater, Aurélien Bellet, and Jan Ramon. "An accurate, scalable and verifiable protocol for federated differentially private averaging". In: *Mach. Learn.* 111.11 (2022), 4249–4293. ISSN: 0885-6125. DOI: 10.1007/s10994-022-06267-9. URL: https://doi.org/10.1007/s10994-022-06267-9.

[Sca+17] Kevin Scaman et al. "Optimal Algorithms for Smooth and Strongly Convex Distributed Optimization in Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 3027–3036. URL: https://proceedings.mlr.press/v70/scaman17a.html.

[Sch16] Bruce Schneier. *Data and Goliath*. New York, NY: WW Norton, Feb. 2016.

[SCS13] Shuang Song, Kamalika Chaudhuri, and Anand D. Sarwate. "Stochastic gradient descent with differentially private updates". In: *2013 IEEE Global Conference on Signal and Information Processing* (2013), pp. 245–248. URL: https://api.semanticscholar.org/CorpusID:8085841.

[SGSJ21] Hossein Shokri Ghadikolaei, Sebastian Stich, and Martin Jaggi. "LENA: Communication-Efficient Distributed Learning with Self-Triggered Gradient Uploads". In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, 2021, pp. 3943–3951. URL: https://proceedings.mlr.press/v130/shokri-ghadikolaei21a.html.

[Sha+21] Fanhua Shang et al. "Differentially Private ADMM Algorithms for Machine Learning". In: *IEEE Transactions on Information Forensics and Security* 16 (2021), pp. 4733–4745.

[Shi+11] Elaine Shi et al. "Privacy-Preserving Aggregation of Time-Series Data". In: *NDSS*. 2011.

[Shi+14] Wei Shi et al. "On the Linear Convergence of the ADMM in Decentralized Consensus Optimization". In: *IEEE Transactions on Signal Processing* 62.7 (2014), pp. 1750–1761.

[Sho+17] Reza Shokri et al. *Membership Inference Attacks against Machine Learning Models*. 2017. arXiv: 1610.05820 [cs.CR].

[Shu20] Sébastien Shulz. "Félix TRÉGUER, L'utopie déchue : une contre-histoire d'Internet, XV e-XXI e siècle , Paris, Fayard, coll. « Histoire de la pensée. À venir », 2019, 351 p." In: *Réseaux* N° 222.4 (July 2020), 209–212. ISSN: 0751-7971. DOI: 10.3917/res.222.0209. URL: http://dx.doi.org/10.3917/res.222.0209.

[Sil+17] David Silver et al. *Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm*. 2017. arXiv: 1712.01815 [cs.AI].

[Sim15] Tom Simonite. "Probing the dark side of Google's ad targeting system". In: *MIT Technology Review* (2015). URL: https://www.technologyreview.com/2015/07/06/110198/probing-the-dark-side-of-googles-ad-targeting-system/.

[Slo16] Bart van der Sloot. "Legal fundamentalism: Is data protection really a fundamental right?" In: *Data Protection and Privacy: (In)visibilities and Infrastructures*. Springer, 2016, pp. 3–30. DOI: 10.1007/978-3-319-50796-5_1.

[SLP18] Latanya Sweeney, Michael von Loewenfeldt, and A M Perry. "Saying it's Anonymous Doesn't Make It So: Re-identifications of "anonymized" law school data". In: 2018. URL: https://api.semanticscholar.org/CorpusID:125680617.

[Smi+17] Virginia Smith et al. "Federated multi-task learning". In: *Advances in neural information processing systems* 30 (2017).

[Sno19] E. Snowden. *Permanent Record*. Henry Holt and Company, 2019. ISBN: 9781250237248. URL: https://books.google.fr/books?id=0XCcDwAAQBAJ.

[Sol08] Daniel J. Solove. *Understanding Privacy*. GWU Legal Studies Research Paper No. 420, GWU Law School Public Law Research Paper No. 420, Available at SSRN: https://ssrn.com/abstract=1127888. Harvard University Press, 2008.

[SP06] Krishna Sampigethaya and Radha Poovendran. "A Survey on Mix Networks and Their Secure Applications". In: *Proceedings of the IEEE* 94.12 (2006), pp. 2142–2181.

[STU17] Adam Smith, Abhradeep Thakurta, and Jalaj Upadhyay. "Is Interaction Necessary for Distributed Private Learning?" In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, May 2017. DOI: 10.1109/sp.2017.35. URL: http://dx.doi.org/10.1109/SP.2017.35.

[Swe00] Latanya Sweeney. "Simple Demographics Often Identify People Uniquely". Working paper. 2000. URL: http://dataprivacylab.org/projects/identifiability/.

[Swe02a] Latanya Sweeney. "Achieving k-Anonymity Privacy Protection Using Generalization and Suppression". In: *Int. J. Uncertain. Fuzziness Knowl. Based Syst.* 10 (2002), pp. 571–588. URL: https://api.semanticscholar.org/CorpusID:1424892.

[Swe02b] Latanya Sweeney. "K-Anonymity: A Model for Protecting Privacy". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (Oct. 2002), pp. 557–570. ISSN: 0218-4885. DOI: 10.1142/S0218488502001648. URL: https://www.worldscientific.com/doi/abs/10.1142/S0218488502001648 (visited on 05/23/2023).

[Swe15] Latanya Sweeney. "Only You, Your Doctor, and Many Others May Know". In: 2015. URL: https://api.semanticscholar.org/CorpusID:58724889.

[SZ13] Ohad Shamir and Tong Zhang. "Stochastic Gradient Descent for Non-smooth Optimization: Convergence Results and Optimal Averaging Schemes". In: *ICML*. 2013.

[TAM19] Om Thakkar, Galen Andrew, and H. B. McMahan. "Differentially Private Learning with Adaptive Clipping". In: *Neural Information Processing Systems*. 2019. URL: https://api.semanticscholar.org/CorpusID:150373827.

[Tan+17] Jun Tang et al. "Privacy Loss in Apple's Implementation of Differential Privacy on MacOS 10.12". In: *ArXiv* abs/1709.02753 (2017). URL: https://api.semanticscholar.org/CorpusID:28672411.

[Tan+18] Hanlin Tang et al. "$D^2$: Decentralized Training over Decentralized Data". In: *ICML*. 2018.

[Ter+22] Jean Ogier du Terrail et al. "Flamby: Datasets and benchmarks for cross-silo federated learning in realistic healthcare settings". In: *Advances in Neural Information Processing Systems (NeurIPS)* 35 (2022), pp. 5315–5334.

[TKC24] Florian Tramèr, Gautam Kamath, and Nicholas Carlini. "Position: Considerations for Differentially Private Learning with Large-Scale Public Pretraining". In: *Proceedings of the 41st International Conference on Machine Learning*. Ed. by Ruslan Salakhutdinov et al. Vol. 235. Proceedings of Machine Learning Research. PMLR, 2024, pp. 48453–48467. URL: https://proceedings.mlr.press/v235/tramer24a.html.

[TSB22] Shirin Tavara, Alexander Schliep, and Debabrota Basu. "Federated Learning of Oligonucleotide Drug Molecule Thermodynamics with Differentially Private ADMM-Based SVM". In: *Machine Learning and Principles and Practice of Knowledge Discovery in Databases: International Workshops of ECML PKDD 2021, Virtual Event, September 13-17, 2021, Proceedings, Part II*. Springer. 2022, pp. 459–467.

[VBT17] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. "Decentralized Collaborative Learning of Personalized Models over Networks". In: *AISTATS*. 2017.

[Vin+19] Oriol Vinyals et al. "Grandmaster level in StarCraft II using multi-agent reinforcement learning". In: *Nature* 575.7782 (Oct. 2019), 350–354. ISSN: 1476-4687. DOI: 10.1038/s41586-019-1724-z. URL: http://dx.doi.org/10.1038/s41586-019-1724-z.

[VW19] Karina Vold and Jessica Whittlestone. "Privacy, Autonomy, and Personalised Targeting: rethinking how personal data is used". en. In: Apollo - University of Cambridge Repository, 2019. DOI: 10.17863/CAM.43129. URL: https://www.repository.cam.ac.uk/handle/1810/296083.

[Wan+19] Zhibo Wang et al. "Beyond Inferring Class Representatives: User-Level Privacy Leakage From Federated Learning". In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. 2019, pp. 2512–2520. DOI: 10.1109/INFOCOM.2019.8737416.

[War65] Stanley L. Warner. "Randomized response: a survey technique for eliminating evasive answer bias." In: *Journal of the American Statistical Association* 60 309 (1965), pp. 63–6. URL: https://api.semanticscholar.org/CorpusID:35435339.

[WB90] Samuel D. Warren and Louis D. Brandeis. "The right to privacy". In: *Harvard Law Review* 4.5 (1890), pp. 193–220. URL: http://faculty.uml.edu/sgallagher/Brandeisprivacy.htm.

[WM19] Sandra Wachter and Brent Mittelstadt. "A Right to Reasonable Inferences: Re-Thinking Data Protection Law in the Age of Big Data and AI". In: *Columbia Business Law Review* 2 (2019). Available at SSRN: https://ssrn.com/abstract=3248829. DOI: 10.2139/ssrn.3248829.

[WO12] Ermin Wei and Asuman E. Ozdaglar. "Distributed Alternating Direction Method of Multipliers". In: *Proceedings of the 51th IEEE Conference on Decision and Control (CDC)*. 2012, pp. 5445–5450.

[WO13] Ermin Wei and Asuman E. Ozdaglar. "On the O(1/k) Convergence of Asynchronous Distributed Alternating Direction Method of Multipliers". In: *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. 2013.

[Woo29] Virginia Woolf. *A Room of One's Own*. London, UK: Hogarth Press, 1929.

[WYX17] Di Wang, Minwei Ye, and Jinhui Xu. "Differentially Private Empirical Risk Minimization Revisited: Faster and More General". In: *NeurIPS*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017.

[XB04] Lin Xiao and Stephen Boyd. "Fast linear iterations for distributed averaging". In: *Systems and Control Letters* 53 (2004), pp. 65–78.

[XZW20] Jie Xu, Wei Zhang, and Fei Wang. *A(DP)2SGD: Asynchronous Decentralized Parallel Stochastic Gradient Descent with Differential Privacy*. Tech. rep. arXiv:2008.09246, 2020.

[XZW21] Jie Xu, Wei Zhang, and Fei Wang. "A$(DP)^2$SGD: Asynchronous Decentralized Parallel Stochastic Gradient Descent with Differential Privacy". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).

[Yak+22] Yauhen Yakimenka et al. "Straggler-Resilient Differentially-Private Decentralized Learning". In: *2022 IEEE Information Theory Workshop (ITW)* (2022), pp. 708–713.

[Yin+21] Bicheng Ying et al. "Exponential Graph is Provably Efficient for Decentralized Deep Training". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer et al. 2021.

[Zig+18] Matteo Zignani et al. *The structure of a decentralized online social network: Mastodon*. NetSci. Poster. June 2018. URL: https://hal.science/hal-02444909.

[ZKL18] Xueru Zhang, Mohammad Mahdi Khalili, and M. Liu. "Improving the Privacy and Accuracy of ADMM-Based Distributed Algorithms". In: *ICML*. 2018.

[ZL22] Shenglong Zhou and Geoffrey Ye Li. *Federated Learning via Inexact ADMM*. Tech. rep. arXiv:2204.10607, 2022.

[ZLH19] Ligeng Zhu, Zhijian Liu, and Song Han. "Deep Leakage from Gradients". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/60a6c4002cc7b29142def8871531281a-Paper.pdf.

[ZLS24] Sajjad Zarifzadeh, Philippe Liu, and Reza Shokri. "Low-Cost High-Power Membership Inference Attacks". In: *Forty-first International Conference on Machine Learning*. 2024. URL: https://openreview.net/forum?id=sT7UJh5CTc.

[ZMB20] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. *iDLG: Improved Deep Leakage from Gradients*. arXiv:2001.02610. 2020.

[Zub19] Shoshana Zuboff. *The age of surveillance capitalism*. en. London, England: Profile Books, Sept. 2019.

[ZZ17] Tao Zhang and Quanyan Zhu. "Dynamic Differential Privacy for ADMM-Based Distributed Classification Learning". In: *IEEE Transactions on Information Forensics and Security (TIFS)* 12.1 (2017), pp. 172–187.