Towards Decentralization, Asynchrony, Privacy and Personalization in Federated Learning

Mathieu Even

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'École Normale Supérieure

Towards Decentralization, Asynchrony, Privacy and Personalization in Federated Learning

Asynchronie, décentralisation, vie privée et personnalisation en apprentissage fédéré

Soutenue par

Mathieu Even

Le 27/06/2024

Ecole doctorale n° 386

Ecole doctorale des sciences mathématiques de Paris centre

Spécialité

Mathématiques

Composition du jury :

Anne-Marie, Kermarrec Professor, EPFL	*Présidente du Jury*
Martin Jaggi Professor, EPFL	*Rapporteur*
Giovanni, Neglia DR, Inria Sofia-Antipolis	*Rapporteur*
Eric, Moulines Professeur, Ecole Polytechnique	*Examinateur*
Laurent, Massoulié DR, Inria Paris	*Directeur de thèse*

Inria

Towards Decentralization, Asynchrony, Privacy and Personalization in Federated Learning

Mathieu Even

Inria, Département d'Informatique de l'ENS
PSL Research University, Paris, France

PhD thesis under the supervision of Laurent Massoulié
September 2021 – June 2024

September 9, 2024

Abstract The advent of modern Machine Learning (ML) models necessitates vast amounts of data and computational power to facilitate accurate predictions. Consequently, these models undergo training in a distributed manner, wherein numerous compute units are employed, and the voluminous training data, too extensive to be centralized on a single device, is decentralized across users' devices.

This thesis delves into various aspects concerning the distributed training of models. Given the escalating scale of computational and data requirements, we initially concentrate on asynchronous training and decentralized methodologies, which confer robustness as the scale of the problem expands. Subsequently, we explore the inherent privacy amplification of decentralized learning and delve into model personalization—an intricate scenario wherein users may not share identical objectives yet necessitate collaboration.

Keywords: Optimization, statistics, federated, distributed, collaborative, decentralized

Bref résumé Les modèles d'apprentissage automatique modernes exigent d'énormes quantités de données et de puissance de calcul pour effectuer des prédictions précises. Par conséquent, ces modèles sont entrainés de manière distribuée: un grand nombre d'unités de calcul sont nécessaires, et les données utilisées pour l'entraînement, étant trop vastes pour être centralisées sur une seule machine, sont stockées de manière décentralisée par les utilisateurs.

Cette thèse explore différentes problématiques liées à l'entraînement distribué de modèles. En raison de l'échelle grandissante des ordres de grandeurs des besoins en calcul et en données nécessaires, nous examinons d'abord les méthodes d'entraînement asynchrones et le cadre décentralisé. Ensuite, nous abordons les questions liées à la protection des données, naturellement amplifiées dans le cadre décentralisé, et la personnalisation des modèles, un scénario plus complexe dans lequel les utilisateurs ne partagent pas nécessairement le même objectif mais doivent néanmoins collaborer.

Mots-clés: Optimisation, statistiques, fédéré, distribué, collaboratif, décentralisé

Summary

Modern machine learning models demand substantial computational power and extensive training data, thus requiring distributed approaches for their training. The use of multiple compute units amplifies processing capabilities, while the decentralization of training data, often held by users, addresses challenges such as data size surpassing the memory capacity of individual computers or containing sensitive information that users are hesitant to share.

In this thesis, we delve into distributed and collaborative learning from both optimization theory and statistical perspectives. We study the convergence properties of existing distributed algorithms and address ways to enhance their efficiency, their privacy properties, and their statistical accuracy.

The first part of this thesis focuses on asynchronous training methods. In synchronous optimization, compute machines all perform computations and communications at the same time. If these machines have heterogeneous compute speeds, faster workers will have to wait for slower ones: this is the *straggler problem*. An alternative to this is asynchronous optimization, where updates on the model are performed whenever possible, without waiting for other nodes, leading to more robustness to large-scale training over many compute nodes and faster algorithms. We study both centralized and decentralized asynchronous optimization algorithms and provide quantitative asynchronous speedups. For asynchronous decentralized optimization, we provide a general continuous-time framework (referred to as *continuized* framework) to develop asynchronous algorithms that benefit from accelerated communications and robustness to delayed updates. In the centralized (*i.e.*, in the presence of a central unit) setting, we study the prevalent asynchronous algorithm and show that provided a slight modification, it is always faster than its synchronous counterpart.

The second part of the thesis investigates privacy-preserving mechanisms for decentralized learning and personalization strategies. We analyze the privacy properties of decentralized learning algorithms and introduce mechanisms for decentralized optimization that benefit from decreasing pairwise privacy leaks between nodes as their distance increases in a communication graph. Additionally, we explore variations of vanilla stochastic optimization algorithms with Markovian noise, encompassing applications such as decentralized algorithms with a random-walker that runs through the communication graph and stochastic approximation problems related to online identification systems. We then delve into model personalization in collaborative learning, exploring inherent statistical limits, statistically optimal algorithms, and personalization strategies based on user or model structural assumptions.

Finally, related to questions that are not central to distributed learning, but to optimization of deep networks and statistics in general, the last part of this thesis includes sets of results on the implicit bias of stochastic gradient descent at the edge of stability for simple non-convex models, as well as concentration results for random tensors.

Résumé

Les modèles d'apprentissage automatique contemporains requièrent une puissance de calcul considérable et une quantité de données d'entraînement de grande envergure, parfois étendues à l'ensemble d'Internet, et sont ainsi entrainés de manière distribuée. Cette approche implique l'utilisation de multiples unités de calcul pour accroître les capacités d'entrainement, tandis que la décentralisation des données, souvent détenues par les utilisateurs, répond à des défis tels que la taille des données dépassant la capacité mémoire des ordinateurs individuels ou la présence d'informations sensibles.

Dans cette thèse, nous explorons l'apprentissage distribué et collaboratif sous les angles des théories de l'optimisation et des statistiques. Nous examinons les propriétés de convergence des algorithmes distribués existants, explorons des stratégies pour améliorer leur efficacité, leurs capacités à amplifier la confidentialité des données, et analysons leur précision statistique.

La première partie de cette thèse se concentre sur les méthodes d'entraînement asynchrones. Dans le cadre de l'optimisation synchrone, toutes les machines effectuent des calculs et des communications simultanément. Toutefois, dans le cas de vitesses de calcul hétérogènes, les travailleurs plus rapides doivent attendre les plus lents. Une alternative consiste en l'optimisation asynchrone, où les mises à jour du modèle sont effectuées sans attendre les autres nœuds, garantissant ainsi une plus grande robustesse pour l'entraînement à grande échelle sur de nombreux nœuds de calcul et des algorithmes plus rapides. Nous examinons à la fois les algorithmes d'optimisation asynchrone centralisés et décentralisés et fournissons des accélérations asynchrones quantitatives précises. Pour l'optimisation asynchrone décentralisée, nous proposons un cadre général en temps continu (appelé cadre "continuisé") pour développer des algorithmes bénéficiant de communications accélérées et d'une robustesse accrue aux mises à jour retardées. Dans le cadre centralisé, nous démontrons qu'un algorithme asynchrone prédominant, moyennant une légère modification, est toujours plus rapide que son homologue synchrone.

La seconde partie de cette thèse étudie les mécanismes de préservation de la confidentialité pour l'apprentissage décentralisé et les stratégies de personnalisation. Nous analysons et introduisons des mécanismes de préservation de la confidentialité des données d'entrainement pour l'optimisation décentralisée. Nos algorithmes bénéficient de manière quantitative du cadre décentralisé, ce qui entraîne une diminution des fuites de confidentialité entre paires de nœuds dans un graphe de communication à mesure que la distance entre ces nœuds augmente. En lien avec les cadres centralisés et décentralisés, nous explorons des variations des algorithmes classiques d'optimisation stochastique avec bruit markovien. Nous examinons également la personnalisation des modèles dans un contexte d'apprentissage collaboratif, en explorant des algorithmes statistiquement optimaux et des stratégies de personnalisation basées sur des hypothèses structurelles liées aux utilisateurs ou aux modèles.

Enfin, la dernière partie de cette thèse comprend des ensembles de résultats sur le biais implicite de la descente de gradient stochastique au bord de la stabilité pour des modèles non convexes simples, ainsi que des résultats de concentration pour des tenseurs aléatoires. Ces résultats abordent des problématiques périphériques à l'apprentissage distribué, mais pertinentes pour l'optimisation des réseaux profonds et les statistiques de manière générale.

Remerciements

Merci avant tout à Laurent pour ton encadrement tout au long de cet expérience qu'a été la thèse, pour m'avoir guidé parfaitement tout en me laissant une dose de liberté. Merci pour ta bienveillance, ta flexibilité, et pour tout ce que tu m'as apporté scientifiquement et humainement. Tu es et resteras une source d'inspiration pour moi.

Je remercie Martin Jaggi et Giovanni Neglia d'avoir accepté d'être rapporteurs de cette thèse. Merci Giovanni pour toutes les très nombreuses remarques sur tout le manuscrit. Merci également à Anne-Marie Kermarrec et Eric Moulines d'avoir constitué mon jury.

Merci évidemment à tout l'équipe Argo (ex-Dyogene), qui a égayé ces années de thèse. En particulier le bureau C321 ("Bureau des boloss") qui a formé et continuera à former de forts joyeux lurons. Merci à vous Luca, Eric, Bastien, Matthieu, Jakob, rigoler et sourire de bon matin avec vous dès l'arrivée au labo n'a pas été désagréable, c'est le moins que l'on puisse dire. En tout cas, on nous le souhaite. Merci à Bertille pour les innombrables pauses tisanes. Merci à toute l'administration, en particulier Hélène (tu nous manques!) et Marina. Merci aussi bien sûr aux nombreux autres amis de l'Inria, avec qui partager grimpes, courses, ragùs et autres goûters a été un vrai bonheur.

Merci à tous les autres amis avec qui j'ai pu partager des moments forts lors de ces dernières années. Et pour finir merci à toute ma famille, papa-maman-arnaud-bertrand, sans oublier Swallie et Bouboule.

Contents

Contributions and outline

We here provide an overall summary of this thesis and its main contributions, by briefly introducing each chapter. If not explicitly stated otherwise, the contributions of this thesis are the author's own. The five parts of this thesis are independent of each other. Within each part, chapters can be read independently from each other. Each chapter is introduced and if necessary placed into the context of preceding chapters with specific related works.

Chapter 1. In this opening chapter, we introduce the Machine Learning concepts used throughout the manuscript and motivate several questions related to distributed learning. In particular, we start from the basis of supervised learning: we relate the computation of an estimator to optimization problems and optimization theory, and the generalization performances of this estimator to statistical learning theory. This leads us to introduce basic first-order and stochastic first-order optimization methods, and their generalization to the distributed centralized and decentralized paradigms. We then focus on the settings related to the contributions of this thesis that we introduce in detail in the distributed learning framework: asynchronous optimization, personalization, and differentially private learning.

Asynchronous optimization. In Part I, we delve into asynchronous optimization, and our contributions in this direction are divided into four chapters.

Chapter 2 introduces a novel continuous-time view on Nesterov accelerated scheme, a close variant of this celebrated algorithm that takes gradient steps at random continuous times. This new process, referred to as the continuized acceleration, benefits from the best of both continuous and discrete worlds (respectively analysis-friendliness and easy implementations), and has applications in decentralized asynchronous optimization. Indeed, accelerating randomized gossip "à la" Nesterov requires strong synchronization between nodes, which our continuized variant bypasses. This chapter is based on the paper *Continuized accelerations of deterministic and stochastic gradient descents, and of gossip algorithms* [EBB+21], published at NeurIPS 2021 and written with Raphaël Berthier, Francis Bach, Nicolas Flammarion, Hadrien Hendrikx, Pierre Gaillard, Laurent Massoulié and Adrien Taylor.

Chapter 3 studies the *Asynchronous SGD* algorithm, a natural asynchronous version of Minibatch SGD in the centralized setting. We show that in centralized optimization, comparing the obtained runtimes of Minibatch SGD and its asynchronous counterpart, the optimizer *always* benefits from asynchrony, with an explicit quantitative asynchronous speedup. This chapter is based on the paper *Asynchronous SGD beats minibatch SGD under arbitrary delays* [MBEW22] published at NeurIPS 2022 and written with Konstantin Mishchenko, Francis Bach and Blake Woodworth. Konstantin initiated this project. The author's contribution was to introduce the delay-dependent stepsizes and to prove Theorem 3.2 (main result under general assumptions).

Chapter 4 studies asynchronous decentralized optimization with heterogeneous communication and computation delays, with a focus on delayed gossip algorithms. This chapter introduces and studies asynchronous algorithms in the continuized framework introduced in Chapter 2. Our delayed randomized gossip algorithm, and its variant for decentralized optimization, consist of pairwise delayed communication updates at random times in a communication graph, combined with local asynchronous gradient steps. We explicit a quantita-

tive asynchronous speedup for delayed randomized gossip that generalizes the asynchronous speedup aforementioned for asynchronous SGD. This chapter is based on the paper *Asynchronous speedup in decentralized optimization* [EHM21b], under review for journal publication, and written with Hadrien Hendrikx and Laurent Massoulié.

Chapter 5 generalizes the analysis of Asynchronous SGD (Chapter 3) to the decentralized setting, providing general results for an asynchronous version of the Decentralized SGD algorithm. Chapter 5 mainly focuses on asynchronous computations and assumes non-delayed communications, unlike Chapter 4. We show that the computational asynchronous speedup of Chapter 3 holds under general decentralized communication assumptions if there are no communication delays. Furthermore, under communication delays that can be heterogeneous, we provide asynchronous communication schemes inspired by the theory of Loss Networks for telecommunication networks, under which the assumptions made in our convergence results hold. This chapter is based on the paper *Asynchronous SGD on Graphs: a Unified Framework for Asynchronous Decentralized and Federated Optimization* [EKM23], published at AISTATS 2024 and written with Anastasia Koloskova and Laurent Massoulié.

Differentially private and personalized learning. In Part II, we study privacy-preserving mechanisms for decentralized learning and personalization strategies.

Chapter 6 studies the natural differential privacy amplification properties of decentralization. We introduce and analyze *Muffliato*, a privacy-preserving mechanism for the decentralized mean estimation problem. This mechanism consists of Gaussian noise injection followed by gossip communications. We introduce a relaxation of Differential Privacy that takes into account the positions of nodes in a decentralized communication graph and shows that Muffliato increases the privacy guarantees as the distance between nodes in the graph increases. We then extend our algorithm to the decentralized optimization setting, replacing the gossip communications of decentralized SGD with our Muffliato mechanism. This chapter is based on the paper *Muffliato: Peer-to-Peer Privacy Amplification for Decentralized Optimization and Averaging* [CEBM22] that was published at NeurIPS 2022 and written with Edwige Cyffers (co-first author), Aurélien Bellet and Laurent Massoulié. The *Muffliato* project started while meeting Edwige at Neurips in Paris 2021, who introduced the author to differential privacy. The contributions of this thesis author in this project are in the design of the Muffliato mechanism, its extension to decentralization, and the proofs (privacy and utility) included in the paper and this chapter.

Chapter 7 studies a variant of vanilla SGD: stochastic gradient descent under Markov chain sampling schemes, where we relax the *i.i.d.*-ness assumption of the stochastic gradients. The two main applications are *token algorithms* — decentralized optimization algorithms where a "token" performs a random walk on a communication graph while iteratively updating the current model with local stochastic gradient steps using the information of local nodes — and online system identification problems. Token algorithms are known to naturally amplify the differential privacy guarantees through the randomness of the random walker. We provide black-box lower bounds for SGD under Markovian sampling schemes, study Markov-Chain SGD under various regularity assumptions, and introduce MC-SAG a variance-reduced version of MC-SGD, that benefits from improved convergence guarantees. This chapter is based on the paper *Stochastic Gradient Descent under Markovian Sampling Schemes* [Eve23] published at ICML 2023.

Chapter 8 studies the min-max information theoretical limits of personalization, under the lens of stochastic optimization. We introduce informational lower bounds for the personalized collaborative learning problem, showing that users cannot benefit from a collaborative speedup larger than their number of neighbors in a similarity graph. We then provide algorithms that are based on some gradient filtering strategies, and that match the introduced lower bounds provided that some similarities are known between agents. If these similarities are not known, we show that they can be estimated and that the induced collaborative

speedup is still optimal. This chapter is based on the paper *On sample optimality in personalized collaborative and federated learning* [EMS22a] published at NeurIPS 2022 and written with Kevin Scaman and Laurent Massoulié.

Chapter 9 studies the problem of collaboratively learning shared structures such as low-rank representations or clustered structures, under a general convex optimization framework. We show that learning such structures is possible, provided that the number of users collaboratively participating is large enough, even if their number of samples is small. Our approach is based on classical regularization techniques adapted to collaborative empirical risk minimization. This chapter is based on a work under review, written with Laurent Massoulié.

Additional contributions of this thesis. The last part of this thesis contains two chapters, that have no direct link to distributed or collaborative learning. We do not provide the proofs of this part in the manuscript due to their weaker relation to the broader subject of the thesis.

Chapter 10 provides uniform concentration inequalities for non-isotropic random tensors with non-linearities. Our concentration inequalities have applications for instance in empirical mean minimization, where uniform concentrations of Hessians are often required. Related to distributed learning, our inequalities provide sharper results for statistical preconditioning, a technique that reduces communication costs. This chapter is based on *Concentration of Non-Isotropic Random Tensors with Applications to Learning and Empirical Risk Minimization* [EM21], published at COLT 2021 and written with Laurent Massoulié.

Chapter 11 studies the implicit bias of stochastic gradient descent over diagonal linear networks, a simple non-convex linear neural network model that can be analyzed. While the optimization side of this thesis aimed at proving that we could minimize the empirical risk efficiently, this does not guarantee that the obtained model has good generalization properties. It has however experimentally been observed that training using stochastic gradient updates leads to models that generalize well: they have a beneficial implicit bias. We completely characterize the solution found by SGD when training diagonal linear networks. Depending on the initialization scale, the noise level, and the stepsize, the limiting model found by stochastic gradient descent can perfectly recover sparse solutions. This chapter is based on *(S) GD over Diagonal Linear Networks: Implicit Regularisation, Large Stepsizes and Edge of Stability* [EPGF23], published at NeurIPS 2023 and written with Scott Pesme, Nicolas Flammarion, and Suriya Gunasekar.

Chapter 1

Introduction to Distributed Learning

In the realm of Machine Learning (ML), the fundamental objective is to derive predictions from existing knowledge. Common ML tasks include addressing linear regression problems, wherein a model is developed to fit a linear pattern through given data points, or tackling classification problems, such as discerning between categories like identifying images as either cats or dogs based on prior instances. Practical examples of ML applications extend to domains like weather forecasting for the upcoming day, where predictions are formulated by using historical weather data spanning decades with the current weather and climate conditions. Similarly, online recommender systems rely on users browsing histories and preferences to offer personalized suggestions, illustrating how ML seamlessly integrates into daily experiences for many individuals.

To address these challenges, predictive models undergo a process of *training* using available data, referred to as the *training set*. Within a given class of potential models, the model demonstrating optimal performance on the training set is typically selected. Subsequently, the aim is for the trained prediction model to demonstrate accuracy when applied to new, unseen data. The design of ML prediction models encompasses four fundamental components: *(i)* assembling a sufficiently expressive training set that aligns with the constraints of the problem, *(ii)* formulating a class of models within which the prediction model will be chosen, ensuring they possess desired properties, *(iii)* efficiently selecting a model from this class that exhibits superior performance on the training data, and *(iv)* guaranteeing the chosen model's efficacy when applied to previously unseen data.

This thesis primarily concentrates on *(iii)*, with some notable contributions to *(iv)*, emphasizing the design of efficient training methods while ensuring robust *generalization* to unseen data within stylized scenarios. The focal point is on comprehending contemporary challenges, particularly in the context of recent ML breakthroughs. Noteworthy achievements, such as language models adept at solving diverse tasks derived from textual inputs [BCE+23], the generation of highly realistic videos [Ope24], and the application of generative models to drug discovery [SSC+18], have been made possible by advancements in architecture, notably the introduction of new structures like transformers [VSP+17], augmented computational power, and the development of efficient training methodologies—a core focus of our investigation.

Many prevalent training methodologies hinge on *stochastic optimization* [BCN18], demonstrating a remarkable capacity to scale effectively with the burgeoning size of modern models. This scalability encompasses both the dimension d of the optimization variable (i.e., the number of model parameters) and the volume of training data, quantified by the number of data samples m in use. Given that d and m regularly reach the scale of hundreds or thousands of billions [Cc22, Tc23], the imperative is growing for the adoption of distributed optimization algorithms. Such algorithms are essential for handling the extensive scale of modern ML models, with federated learning [MMR+17] representing a specific instance of parallel and distributed optimization wherein communications and synchrony are restricted, and data sharing may be limited.

In the introduction of this manuscript, we will introduce the different flavors of distributed

learning we consider in this thesis together with our contributions to the field.

1.1. Basics of learning and optimization theories

In this chapter, we frame ML as an optimization problem and introduce basic concepts of optimization and learning theories. Note that we only consider supervised learning settings in the introduction.

The first component of ML is the training data, which we write as $\{\xi_i = (a_i, b_i)\,,\, i \in [m]\}$, where $m \in \mathbb{N}$ is the number of data samples available, a_i's are the inputs that live in a space $\mathbb{A}$, and b_i's are their corresponding labels or predictions, that live in $\mathbb{B}$. Given a new $a \in \mathbb{A}$, the goal is to predict its associated label b. In ML, the learning task is formulated as learning a predictive model $\phi^\star : \mathbb{A} \to \mathbb{B}$ such that for almost all new inputs $a \in \mathbb{A}$ and associated label b, we $\phi^\star(a) \approx b$, up to some problem-dependent errors that might be irreducible. This function $\phi^\star$ is then assumed to lie in a (or in the vacinity of a) set $\mathcal{H} \subset \mathbb{B}^{\mathbb{A}}$ that is parameterized by $x \in \mathbb{R}^d$, in the sense that $\mathcal{H} = \left\{\varphi(x;\cdot)\,,\, x \in \mathbb{R}^d\right\}$, where $\varphi : \mathbb{R}^d \times \mathbb{A} \to \mathbb{B}$. The primary goal of the training phase is thus to find the right $x \in \mathbb{R}^d$ *i.e.*, a *model* x that satisfies $\varphi(x,\cdot) \approx \phi^\star$.

Finding x such that $\varphi(x,\cdot) \approx \phi^\star$ on the whole set $\mathbb{A}$ is impossible without any structural assumption, since we only have access to m data samples $\{\xi_i = (a_i, b_i)\,,\, i \in [m]\}$. We will thus fall back to finding $x \in \mathbb{R}^d$ such that for all $i \in [m]$, we have $\varphi(x, a_i) \approx b_i$; the meaning of the approximation $\approx$ may differ depending on the problem. This can be formulated as a minimization problem: given a *loss function* $\ell : \mathbb{B} \times \mathbb{B} \to \mathbb{R}^+$, $\ell(\hat{b}, b)$ is the loss incurred when predicting $\hat{b}$ instead of b: $\ell(\hat{b}, b) = 0$ if $\hat{b} = b$, and increases when $\hat{b}$ is far from b. A candidate for x then is, assuming its existence:

$$x^\star_{\mathrm{ERM}} \in \operatorname*{arg\,min}_{x \in \mathbb{R}^d} \left\{ \mathcal{L}(x) = \frac{1}{m} \sum_{i=1}^{m} \ell\left(\varphi(x, a_i), b_i\right) \right\}. \tag{ERM}$$

This model candidate $x^\star_{\mathrm{ERM}}$ is often referred to as the *empirical risk minimizer* (ERM) and $\mathcal{L}$ as the empirical risk.

Example 1.1.1. *The two following examples are the most commonly studied in ML theory.*

1. Binary classification. *In binary classification, the inputs a_i may lie in a subset of $\mathbb{R}^p$, while their labels b_i are in $\{-1, 1\}$: they are binary. In linear binary classification, one seeks to find $x \in \mathbb{R}^d$ such that the sign of the scalar product $\langle a, x\rangle$ ($p = d$ in this case) predicts the associated label b with good accuracy. The associated losses are the binary loss $\ell(\langle a, x\rangle, b) = \mathbb{1}_{\{\mathrm{sign}(\langle a,x\rangle) \neq b\}}$ or the logistic loss $\ell(\langle a, x\rangle, b) = \ln\left(1 + \exp\left(-b\langle a, x\rangle\right)\right)$.*

2. Linear regression. *In linear regression, the inputs lie in $\mathbb{R}^d$ and one seeks $x \in \mathbb{R}^d$ such that $b \approx \langle a, x\rangle$: there is a linear relation between inputs a and associated b's. The associated loss is the ℓ^2 loss, defined as $\ell(\langle a, x\rangle, b) = \frac{1}{2}(\langle a, x\rangle - b)^2$.*

Note that instead of minimizing the empirical loss, depending on the structure of the problem, one may choose to minimize the penalized empirical loss $\mathcal{L}(x) + \lambda\Omega(x)$ for some penalty Ω and regularization parameter λ. Such penalties may then lead to either faster optimization or stronger statistical guarantees.

1.1.1. Minimizing the empirical risk: gradient descent and stochastic gradient descent

The first question that arises now is: how do we effectively minimize $\mathcal{L}$ and approximate the empirical minimizer $x^\star_{\mathrm{ERM}}$ defined in (ERM)? While without any assumption, optimization problems such as (ERM) can be arbitrarily hard to solve, we will consider sets of assumptions (see Section 1.6) that make iterative methods such as *gradient descent* effective. Starting from some initial model $x_0 \in \mathbb{R}^d$, gradient descent aims at iteratively improving the

model by taking steps in directions where $\mathcal{L}$ decreases the most. This can be thought of as simply going down the valley in the steepest direction when mountaineering and looking for the lowest point in the area.

Gradient descent. More formally, assuming that $\mathcal{L}$ is *differentiable*, gradient descent performs the following iterations, for $k \geq 0$:

$$x_{k+1} = x_k - \eta_k \nabla \mathcal{L}(x_k)\,, \tag{GD}$$

where $\eta_k > 0$ is the *stepsize* or *learning rate* (depending on the terminology used), and $\nabla \mathcal{L}(x_k)$ is the gradient of $\mathcal{L}$ at x_k, that points towards larger values of $\mathcal{L}$. Taking a step $-\eta_k \nabla \mathcal{L}(x_k)$ in the opposite direction should thus decrease the value of $\mathcal{L}$. Gradient descent is a *first-order method*: it requires the knowledge and the computation of first-order quantities related to $\mathcal{L}$ (its gradient).

In the general case, the sequence generated by (GD) is not ensured to converge nor to decrease the loss for arbitrary stepsizes. We recall in a latter section (Section 1.6) sets of assumptions under which studying gradient descent is possible: (strong-)convexity of $\mathcal{L}$, smoothness, Lipschitzness, Polyak-Łojasiewicz assumptions, ... These assumptions may not be necessarily verified in certain applications (*e.g.*, deep learning applications); optimization theory however seeks to derive stylized frameworks under which the aforementioned algorithm and its variants can be studied.

Proposition 1.1.1. *Assume that $\mathcal{L} : \mathbb{R}^d \to \mathbb{R}$ is convex (for all $x, y \in \mathbb{R}^d$, $\mathcal{L}(x) - \mathcal{L}(y) \geq \langle \nabla \mathcal{L}(y), x - y \rangle$) and $L-$smooth (its gradient is $L-$Lipschitz). Assume the existence of $x^\star_{\mathrm{ERM}} \in \mathbb{R}^d$ a minimizer of $\mathcal{L}$. Then, for constant stepsizes $\eta_k \equiv \frac{1}{2L}$, for all $K > 0$ we have:*

$$\mathcal{L}\left(\frac{1}{K}\sum_{k<K} x_k\right) - \mathcal{L}(x^\star_{\mathrm{ERM}}) \leq \frac{2LB^2}{K}\,,$$

where B^2 is an upper bound on $\|x_0 - x^\star_{\mathrm{ERM}}\|^2$.

Proof. Denote $x^\star = x^\star_{\mathrm{ERM}}$ to ease notations. Developing $e_{k+1} = \|x_{k+1} - x^\star\|^2$, we have that $e_{k+1} = e_k - 2\eta_k \langle \nabla \mathcal{L}(x_k), x_k - x^\star \rangle + \eta_k^2 \|\nabla \mathcal{L}(x_k)\|^2$. First, using convexity, $-2\eta_k \langle \nabla \mathcal{L}(x_k), x_k - x^\star \rangle \leq -2\eta_k (\mathcal{L}(x_k) - \mathcal{L}(x^\star))$, and using smoothness, $\|\nabla \mathcal{L}(x_k)\|^2 \leq 2L(\mathcal{L}(x_k) - \mathcal{L}(x^\star))$ (see Section 1.6). Thus, $e_{k+1} - e_k \leq -2\eta_k(1 - L\eta_k)(\mathcal{L}(x_k) - \mathcal{L}(x^\star))$. Summing over $k < K$:

$$\sum_{k<K} 2\eta_k \left(1 - \frac{L\eta_k}{2}\right)(\mathcal{L}(x_k) - \mathcal{L}(x^\star)) \leq e_0 - e_K\,.$$

For stepsizes $\eta_k \equiv \frac{1}{2L}$, this leads to $\frac{1}{K}\sum_{k<K}(\mathcal{L}(x_k) - \mathcal{L}(x^\star)) \leq \frac{2Le_0}{K}$, and to the desired result by convexity.

□

Stochastic gradient descent. Let ℓ' be the derivative of ℓ with respect to its first variable. The gradient of $\mathcal{L}$ writes as

$$\nabla \mathcal{L}(x_k) = \frac{1}{m}\sum_{i=1}^{m} \ell'\left(\varphi(x, a_i), b_i\right) \nabla_x \varphi(x_k, a_i)\,,$$

where $\nabla_x \varphi$ is the gradient of φ with respect to its first variable. Computing $\nabla \mathcal{L}(x_k)$ has a complexity of $m\times$complexity of computing $\ell'\left(\varphi(x, a_i), b_i\right) \nabla_x \varphi(x_k, a_i)$. This complexity in m can be paralleled, since each $\ell'\left(\varphi(x, a_i), b_i\right) \nabla_x \varphi(x_k, a_i)$ can be computed on a different device.

However, m can be quite large in modern applications. For instance, the MNIST dataset consists of 60000 digits, while large language models are trained on billions of examples. Hence, computing $\nabla\mathcal{L}$ at each iteration can become highly ineffective.

Thus, instead of using all available samples at each iteration, one may alternatively use *stochastic gradient descent* (SGD) instead of (GD):

$$x_{k+1} = x_k - \eta_k g_k\,, \tag{SGD}$$

where g_k is an *unbiased* estimate of $\nabla\mathcal{L}(x_k)$ (*i.e.*, $\mathbb{E}\left[g_k|x_0,\dots,x_k\right] = \nabla\mathcal{L}(x_k)$) independent of the previous iterates. Such estimates of the gradients are usually much cheaper to compute, the most natural one consisting of choosing only a random subset $\mathcal{B}_k \subset [m]$ sampled uniformly and independently from the past in all subsets of $[m]$ of a given cardinality, to compute $g_k = \frac{1}{|\mathcal{B}_k|}\sum_{i\in\mathcal{B}_k} \ell'\left(\varphi(x_k,a_i),b_i\right)\nabla_x\varphi(x_k,a_i)$, therefore reducing the cost per iteration from proportional to m to proportional to $|\mathcal{B}_k|$ that can be arbitrarily smaller. $|\mathcal{B}_k|$ is usually referred to as the *batch size*. While this comes at the cost of slower convergence in worst-case scenarios, in practice using (SGD) instead of (GD) proves to be much faster. We refer the interested reader to [GG23, GLQ+19] for generic convergence results on the iterates (GD) and (SGD).

Proposition 1.1.2. *Under the assumptions of Proposition 1.1.1, if additionally stochastic gradients satisfy* $\mathbb{E}\left[g_k|\mathcal{F}_k\right] = \nabla\mathcal{L}(x_k)$ *(unbiasedness) and* $\mathbb{E}\left[\|g_k - \nabla\mathcal{L}(x_k)\|^2|\mathcal{F}_k\right] \le \sigma^2$ *(bounded variance) where* $\mathcal{F}_k$ *is the filtration up to iteration* k*, then for all* $K > 0$ *and constant stepsizes* $\eta_k \equiv \min\left(\frac{1}{2L}, \sqrt{\frac{B^2}{K\sigma^2}}\right)$*:*

$$\mathbb{E}\left[\mathcal{L}\left(\frac{1}{K}\sum_{k<K} x_k\right) - \mathcal{L}(x^\star_{\mathrm{ERM}})\right] \le \frac{2LB^2}{K} + 2\sqrt{\frac{\sigma^2B^2}{K}}\,. \tag{1.1}$$

Proof. As in the previous proof, let $x^\star = x^\star_{\mathrm{ERM}}$. We proceed as in the proof of Proposition 1.1.1, but with conditional means. $\mathbb{E}\left[e_{k+1}|\mathcal{F}_k\right] = e_k - 2\eta_k\mathbb{E}\left[\langle g_k, x_k - x^\star\rangle|\mathcal{F}_k\right] + \eta_k^2\mathbb{E}\left[\|g_k\|^2|\mathcal{F}_k\right]$. First, by linearity of the mean and since x_k is $\mathcal{F}_k-$measurable, $\mathbb{E}\left[\langle g_k, x_k - x^\star\rangle|\mathcal{F}_k\right] = \langle\mathbb{E}\left[g_k|\mathcal{F}_k\right], x_k - x^\star\rangle = \langle\nabla\mathcal{L}(x_k), x_k - x^\star\rangle$. Then, using the unbiasedness and the bounded variance, $\eta_k^2\mathbb{E}\left[\|g_k\|^2|\mathcal{F}_k\right] = \eta_k^2\mathbb{E}\left[\|g_k - \nabla\mathcal{L}(x_k)\|^2|\mathcal{F}_k\right] + \eta_k^2\mathbb{E}\left[\|\nabla\mathcal{L}(x_k)\|^2|\mathcal{F}_k\right] \le \eta_k^2\sigma^2 + \eta_k^2\mathbb{E}\left[\|\nabla\mathcal{L}(x_k)\|^2|\mathcal{F}_k\right]$. As in the proof of Proposition 1.1.1, this leads us to, using smoothness and convexity:

$$\mathbb{E}\left[e_{k+1}|\mathcal{F}_k\right] - e_k \le -2\eta_k(1 - L\eta_k)(\mathcal{L}(x_k) - \mathcal{L}(x^\star)) + \eta_k^2\sigma^2\,.$$

Taking the mean and summing:

$$\sum_{k<K} 2\eta_k(1 - L\eta_k)\mathbb{E}\left[\mathcal{L}(x_k) - \mathcal{L}(x^\star)\right] \le e_0 - \mathbb{E}\left[e_K\right] + \sigma^2\sum_{k<K}\eta_k^2\,.$$

For constant stepsizes $\eta_k \equiv \eta \le \frac{1}{2L}$,

$$\frac{1}{K}\sum_{k<K}\mathbb{E}\left[\mathcal{L}(x_k) - \mathcal{L}(x^\star)\right] \le \frac{B^2}{K\eta} + K\sigma^2\eta\,.$$

We then get the desired result by plugging the value of η (obtained by optimizing the RHS) in the above equation. □

1.1.2. Generalization to new data for random models: statistical perspectives and generalization bounds

Now that we have presented the main ideas behind the minimization of $\mathcal{L}$ and the approximation of $x^\star_{\text{ERM}}$, what does "generalizing to new and unseen data" mean? Without any assumption on both the training set (the available data samples $\xi_i = (a_i, b_i)$ for $i \in [m]$) and on unseen data on which we want to generalize on, we cannot ensure any guarantee that our model will perform well. For instance, the training set may only be confined to a small subspace of the whole space, or the unseen data may be drawn in an adversarial way.

That is where statistical learning theory and its framework comes, and makes the assumption that the samples ξ_i are drawn *i.i.d.* from a distribution $\mathcal{D}$, while the "unseen data" is also drawn from $\mathcal{D}$, independently from the training set. We thus deal with random models, to be able to derive generalization guarantees. Thus, the average error on this unseen data writes as:

$$\mathcal{E}(x) = \mathbb{E}\left[\ell(\varphi(x, a), b)\right], \qquad \xi = (a, b) \sim \mathcal{D}\,. \tag{1.2}$$

The generalization error then writes as:

$$\mathcal{E}(x) - \inf_{x'} \mathcal{E}(x')\,, \tag{1.3}$$

and is thus the optimality gap between x and an optimal model that minimizes $\mathcal{E}$. There are however other ways to obtain and quantify generalization guarantees for random models, such as PAC ("probably approximately correct") learning for instance, that we do not consider in this manuscript, where we only consider Equation (1.3) as a generalization criterion.

The generalization error of a model x can then be decomposed into two terms: an empirical optimization error term and a variance error term:

$$\left\{\mathcal{L}(x) - \min_{x'} \mathcal{L}(x')\right\} + 2\left\{\sup_{x'} \left|\mathcal{L}(x') - \mathcal{E}(x')\right|\right\}.$$

The first term can be written as $\mathcal{L}(x) - \mathcal{L}(x^\star_{\text{ERM}})$ where $x^\star_{\text{ERM}}$ minimizes the empirical risk (ERM), and determines how well the empirical risk minimizer is approximated. The second term is a variance term, since for all x' we have $\mathbb{E}\left[\mathcal{L}(x')\right] = \mathcal{E}(x')$, and will decrease as m increases. Due to the supremum taken over all x', under adequate assumptions, this variance term will scale as $\mathcal{O}\left(\frac{d}{m}\right)$.

A "good" prediction model x thus minimizes both optimization error and variance error, which can here be decoupled. However, in the "online" or "one-pass" setting, these can be the same. If m the number of data samples is very large (larger than the number of epochs of (SGD) times the batch size, typically), then each data sample will be seen only once. Consequently, the stochastic gradients g_k are in fact stochastic gradients of the generalization error: $\mathbb{E}\left[g_k | x_0, \dots, x_k\right] = \nabla\mathcal{E}(x_k)$, since the samples used to compute g_k have not been used before and are independent of the previous k steps. Hence, in that case, stochastic gradient descent directly minimizes the generalization error, and there is no bias term. Under similar assumption as for Proposition 1.1.2, we expect the same rate to hold, where this time L is the smoothness of $\mathcal{E}$, σ^2 the variance of new stochastic gradients, and $B^2 \geq \|x_0 - x^\star\|^2$ where $x^\star$ a minimizer $\mathcal{E}$ the generalization error is an optimal model.

Different types of arguments exist to upper bound the generalization error (1.3): stability arguments, Rademacher complexity, etc ... [BE02, BBL04, BB07]. Overall, under adequate assumptions (convexity, smoothness, and random samples), the generalization error would satisfy bounds of the form

$$\mathcal{E}(\hat{x}) - \inf_{x'} \mathcal{E}(x') \leq \sqrt{\frac{Cd}{m}}\,, \tag{1.4}$$

where d is the dimension of the search space, and m the number of samples, and C depends

on the regularity of ℓ, ϕ and the law of the samples ξ. Under strong convexity assumptions of the objective, the rate in Equation (1.4) would be replaced by the faster rate $\frac{C'd}{m}$.

We present a simplified version here for linear regression with the quadratic loss and Gaussian data to provide an example of the formalism used throughout the thesis, and to what we refer to when tackling generalization properties. The proof below — whose main ideas are very classical but that the author nonetheless finds quite nice in that the approach is general and can be applied to any problem — also gives the intuition of where the different factors (dimension d and inverse of the number of samples $1/n$) come from. We refer the interested reader to [Gir21, Wai19] for detailed and pedagogical overviews of high-dimensional statistics techniques.

Proposition 1.1.3. *Assume that data $(a_i, b_i) \sim \mathcal{D}$ is generated as $b_i = \langle x^\star, a_i \rangle + z_i$, where $a_i \sim \mathcal{N}(0, I_d)$ and $z_i \sim \mathcal{N}(0, \sigma^2)$ are independent Gaussians, and $x^\star \in \mathbb{R}^d$ satisfies $\|x^\star\| \leq 1$. Assume that the loss ℓ is the quadratic loss ($\ell(b', b) = \frac{1}{2}(b - b')^2$) and that the function ϕ is linear ($\varphi(x, a) = \langle x, a \rangle$). Then, the generalization error (1.3) writes, for any $x \in \mathbb{R}^d$:*

$$\mathcal{E}(x) - \inf_{x'} \mathcal{E}(x') = \frac{1}{2}\|x - x^\star\|^2 ,$$

and the regularized empirical risk minimizer defined as

$$\hat{x} \in \underset{x \in \mathbb{R}^d \,, \|x\| \leq 1}{\arg\min} \left\{ \mathcal{L}(x) = \frac{1}{m} \sum_{i=1}^{m} \ell\left(\varphi(x, a_i), b_i\right) \right\} .$$

satisfies, with probability $1 - \delta$:

$$\|\hat{x} - x^\star\|^2 \leq C \max\left(\frac{d \ln(n) + \ln(1/\delta)}{n}, \sqrt{\frac{d \ln(n) + \ln(1/\delta)}{n}} \right) ,$$

where $C > 0$ is a numerical constant.

Proof. By definition of $\mathcal{L}$, we have $\mathcal{L}(\hat{x}) \leq \mathcal{L}(x^\star)$. This writes as:

$$\frac{1}{2n} \sum_{i=1}^{n} \langle a_i, \hat{x} - x^\star \rangle^2 \leq \frac{1}{n} \sum_{i=1}^{n} z_i \langle a_i, \hat{x} - x^\star \rangle .$$

Since a_i's are standard centered Gaussians, for fixed $\Delta \in \mathbb{R}^d$, $\mathbb{E}\left[\langle a_i, \Delta \rangle^2\right] = \|\Delta\|^2$. Random variables $\langle a_i, \Delta \rangle^2$ are then independent $\|\Delta\|^2$−subexponential random variables. We can thus use Hanson-Wright inequality [HW71] to bound the deviations of $\frac{1}{n}\sum_{i=1}^{n} \langle a_i, \Delta \rangle^2$ around its mean:

$$\mathbb{P}\left(\left| \frac{1}{n} \sum_{i=1}^{n} \langle a_i, \Delta \rangle^2 - \|\Delta\|^2 \right| > c\|\Delta\|^2 \left(\sqrt{\frac{\lambda}{n}} + \frac{\lambda}{n} \right) \right) \leq 2 \exp\left(-\lambda\right) .$$

where $c > 0$ is some numerical constant. Then, we would like to apply this to $\Delta = \hat{x} - x^\star$. However, this comes with a twist: $\hat{x}$ is not independent from the training set, since it has been constructed as a function of the training set! To use the above deviations, we therefore need to compute *uniform deviations*, which are valid for all $\Delta \in \mathbb{R}^d$.

We will thus resort to a discretization argument. First, note that the above expression in the probability is *homogeneous* in $\|\Delta\|$, so that we can work on the unit sphere. Let $\mathcal{N}_\varepsilon \subset \mathcal{S}$ be an ε−net of minimal cardinality of $\mathcal{S}$, the unit sphere of $\mathbb{R}^d$: for all $x \in \mathcal{S}$, there exists $x' \in \mathcal{N}_\varepsilon$ such that $\|x - x'\| \leq \varepsilon$. We know that $\ln(|\mathcal{N}_\varepsilon|) \leq d \ln(1 + 2/\varepsilon)$ [Dud74]. Using a

union bound and setting $\lambda = d\ln(1+2/\varepsilon)+t$, we have:

$$\mathbb{P}\left(\forall \Delta \in \mathcal{N}_\varepsilon\,,\ \left|\frac{1}{n}\sum_{i=1}^n \langle a_i,\Delta\rangle^2 - 1\right| > c\left(\sqrt{\frac{d\ln(1+2/\varepsilon)+t}{n}} + \frac{d\ln(1+2/\varepsilon)+t}{n}\right)\right) \leq 2\exp(-t)\,.$$

We want to extend this to $\mathcal{S}$. Let $\Delta \in \mathcal{S}$, and $\Delta' \in \mathcal{N}_\varepsilon$ such that $\|\Delta - \Delta'\| \leq \varepsilon$. The function $x \in \mathcal{S} \mapsto \langle a_i, x\rangle^2$ is $L = 2\max_i \|a_i\|^2$ Lipschitz. Thus, $\left|\frac{1}{n}\sum_{i=1}^n \langle a_i,\Delta\rangle^2 - 1\right| \leq \left|\frac{1}{n}\sum_{i=1}^n \langle a_i,\Delta\rangle^2 - 1\right| + L\varepsilon$. For $\varepsilon < \frac{1}{2L}$, we thus have that, with probability $1-2e^{-t}$, for all $\Delta \in \mathcal{S}$,

$$\begin{aligned}\frac{1}{2} - c\left(\sqrt{\frac{d\ln(1+2/\varepsilon)+t}{n}} + \frac{d\ln(1+2/\varepsilon)+t}{n}\right) &\leq \frac{1}{n}\sum_{i=1}^n \langle a_i,\Delta\rangle^2 \\ &\leq \frac{3}{2} + c\left(\sqrt{\frac{d\ln(1+2/\varepsilon)+t}{n}} + \frac{d\ln(1+2/\varepsilon)+t}{n}\right),\end{aligned}$$

and in particular, for $\Delta = \hat{x} - x^\star$ (by renormalizing),

$$\frac{1}{n}\sum_{i=1}^n \langle a_i, \hat{x}-x^\star\rangle^2 \geq \left(\frac{1}{2} - c\left(\sqrt{\frac{d\ln(1+2/\varepsilon)+t}{n}} + \frac{d\ln(1+2/\varepsilon)+t}{n}\right)\right)\|\hat{x}-x^\star\|^2\,.$$

The proof is not over yet. However, the goal of this introduction is only to give a brief overview of techniques, so we will only give a sketch of proof for the remaining of this proof, and refer the interested reader to the books [Gir21, Wai19] for detailed arguments.

First, L is random: we can bound it with high probability using the concentration of χ^2 random variables [Ver18] and a union bound over all n random variables. Then, the noise term $\frac{1}{n}\sum_{i=1}^n z_i\langle a_i, \hat{x}-x^\star\rangle$ is bounded exactly as we did for $\frac{1}{2n}\sum_{i=1}^n \langle a_i,\hat{x}-x^\star\rangle^2$, except that this time this quantity is centered so only deviations appear.

□

1.1.3. Implicit regularization: relation between optimization algorithms and generalization

The stochastic gradient descent algorithm (SGD) [RM51] is the foundational algorithm for almost all neural network training. Though a remarkably simple algorithm, it has led to many impressive empirical results and is a key driver of deep learning. However, the performances of first-order optimization algorithms are quite puzzling from a theoretical point of view as (1) their convergence is highly non-trivial for non-convex models (if the optimization (ERM) is non-convex) and (2) there exist many solutions to (ERM) for the training objective which generalise very poorly [ZBH+17].

To explain this second point, the concept of implicit regularisation has emerged: if overfitting (fitting all training points perfectly) is harmless in many real-world prediction tasks, it must be because the optimization process is *implicitly favoring* solutions that have good generalization properties for the task. The canonical example is overparametrised linear regression with more trainable parameters than number of samples: although there are infinitely many solutions that fit the samples, GD and SGD explore only a small subspace of all the possible parameters. As a result, it can be shown that they implicitly converge to the closest solution in terms of the ℓ_2 distance, and this without explicit regularisation [ZBH+17, GLSS18a], as shown below. Proposition 1.1.4 is an independent contribution of this manuscript, and differs from previous approaches [ZBH+17, GLSS18a] by showing almost

sure convergence instead of directly assuming convergence of the iterates.

Proposition 1.1.4. *Let us consider the quadratic linear regression problem:*

$$\mathcal{L}(x) = \frac{1}{2m} \sum_{i=1}^{m} \left(\langle x, a_i \rangle - b_i \right)^2 ,$$

for inputs $a_i \in \mathbb{R}^d$, model $x \in \mathbb{R}^d$ and labels $b_i \in \mathbb{R}^d$. In the overparametrised regime, there exists some $x \in \mathbb{R}^d$ such that for all $i \in [m]$, we have $\langle x, a_i = b_i$: such a x is an interpolator and $\mathcal{L}(x) = 0$. The overparametrised regime will in most cases always happen for dimensions larger than the number of samples: $d \gg m$.

*Consider computing the empirical risk minimizer (ERM) via stochastic gradient descent as in (SGD) initialized at some $x_0 \in \mathbb{R}^d$ for some small enough constant stepsizes $\eta_k \equiv \eta$. Then, the iterates (x_k) will converge almost surely to some interpolating vector $x^\star_{\ell^2}$ (*i.e.*, such that $\mathcal{L}(x^\star_{\ell^2}) = 0$) characterized by:*

$$x^\star_{\ell^2} \in \operatorname*{arg\,min}_{x^\star \in \mathbb{R}^d \,,\, \forall i \in [m] \,,\, \langle a_i, x^\star \rangle = b_i} \|x_0 - x^\star\|_2^2 ,$$

which is an implicit regularization problem.

Proof. The stochastic gradients write as $g_k = \frac{1}{N} \sum_{i \in \mathcal{B}_k} a_i a_i^\top (x_k - x^\star)$ where $\mathcal{B}_k$ is drawn uniformly at random in the subsets of $[m]$ of size N. Let $x^\star$ be any interpolating vector. Let $L > 0$ be such that for all subsets $\mathcal{B}$ of $[m]$ of size N, we have $\left\| \frac{1}{N} \sum_{i \in \mathcal{B}_k} a_i a_i^\top (x_k - x^\star) \right\|_2^2 \leq \frac{L}{N} \sum_{i \in \mathcal{B}_k} (x_k - x^\star) a_i a_i^\top (x_k - x^\star)$: L is an upper bound on the smoothness of all stochastic functions sampled. We have, using computations similar to those of Proposition 1.1.1:

$$\begin{aligned}
\|x_{k+1} - x^\star\|^2 &= \|x_k - x^\star\|^2 + \eta^2 \left\| \frac{1}{N} \sum_{i \in \mathcal{B}_k} a_i a_i^\top (x_k - x^\star) \right\|_2^2 - \frac{2\eta}{N} \sum_{i \in \mathcal{B}_k} (x_k - x^\star) a_i a_i^\top (x_k - x^\star) \\
&\leq \|x_k - x^\star\|^2 + \frac{\eta^2 L}{N} \sum_{i \in \mathcal{B}_k} (x_k - x^\star) a_i a_i^\top (x_k - x^\star) \\
&\quad - \frac{2\eta}{N} \sum_{i \in \mathcal{B}_k} (x_k - x^\star) a_i a_i^\top (x_k - x^\star) \\
&\leq \|x_k - x^\star\|^2 - \frac{\eta}{N} \sum_{i \in \mathcal{B}_k} (x_k - x^\star) a_i a_i^\top (x_k - x^\star) ,
\end{aligned}$$

for $\eta < 1/L$. Thus, $\|x_{k+1} - x^\star\|^2 \leq \|x_k - x^\star\|^2$ and by induction the sequence (x_k) is bounded. Then, summing as in Proposition 1.1.1, we have that, almost surely:

$$\sum_{k < \infty} \mathcal{L}_k(x_k) \leq \frac{\|x_0 - x^\star\|^2}{\eta} ,$$

where $\mathcal{L}_k(x_k) = \frac{\eta}{N} \sum_{i \in \mathcal{B}_k} (x_k - x^\star) a_i a_i^\top (x_k - x^\star)$. We now would like to show that $\sum_{k < \infty} \mathcal{L}(x_k) < \infty$. We can write

$$\sum_{k < K} \mathcal{L}(x_k) = \sum_{k < K} \mathcal{L}_k(x_k) + M_K ,$$

where $M_K = \sum_{k<K} \mathcal{L}(x_k) - \mathcal{L}_k(x_k)$ is a *martingale*. Now, notice that $M_K = \sum_{k<K} \mathcal{L}(x_k) - \sum_{k<\infty} \mathcal{L}_k(x_k) \geq - \sum_{k<\infty} \mathcal{L}_k(x_k) \geq - \frac{\|x_0 - x^\star\|^2}{\eta}$, since $\sum_{k<K} \mathcal{L}(x_k) \geq 0$ and using what we have proved above. Consequently, M_K is a lower bounded martingale, so that using Doob's first martingale convergence theorem [Doo90] — a lower bounded supermartingale converges

almost surely —, M_K converges almost surely. We can thus conclude that $\sum_{k<K} \mathcal{L}(x_k)$ converges also almost surely, so that $\mathcal{L}(x_k) \to 0$.

Let $x^\star_{\ell^2}$ be (uniquely) defined as:

$$x^\star_{\ell^2} \in \operatorname*{arg\,min}_{x^\star \in \mathbb{R}^d \,,\, \forall i \in [m] \,,\, \langle a_i, x^\star \rangle = b_i} \|x_0 - x^\star\|_2^2 \,,$$

we would like to show that $x_k \to x^\star_{\ell^2}$. Using KKT conditions of the above problem, we have that $x^\star_{\ell^2} - x_0 \in \mathrm{Span}(\mathrm{a}_1, \dots, \mathrm{a}_\mathrm{n})$.

For all $k \geq 0$, decompose x_k as $x_k = x_0 + y_k + z_k$, where $y_k \in \mathrm{Span}(\mathrm{a}_1, \dots, \mathrm{a}_\mathrm{n})$ and $z_k \in \mathrm{Span}(\mathrm{a}_1, \dots, \mathrm{a}_\mathrm{n})^\perp$. Notice that for all k, $g_k \in \mathrm{Span}(\mathrm{a}_1, \dots, \mathrm{a}_\mathrm{n})$, so that $z_k = 0$ for all k. Thus, we have that $x_k - x_0, x^\star_{\ell^2} - x_0 \in \mathrm{Span}(\mathrm{a}_1, \dots, \mathrm{a}_\mathrm{n})$, so that $x_k - x^\star_{\ell^2} \in \mathrm{Span}(\mathrm{a}_1, \dots, \mathrm{a}_\mathrm{n})$. Now, by definition, the matrix $\sum_{i=1}^m a_i a_i^\top$ is invertible on $\mathrm{Span}(\mathrm{a}_1, \dots, \mathrm{a}_\mathrm{n})$: since it is symmetric and positive semi-definite, it is thus positive definite. Hence, there exists $\mu > 0$ such that for all $x \in \mathrm{Span}(\mathrm{a}_1, \dots, \mathrm{a}_\mathrm{n})$, we have $\sum_{i=1}^m x^\top a_i a_i^\top x \geq \mu \|x\|^2$. Thus, since $\mathcal{L}(x_k) = \sum_{i=1}^m (x^\star_{\ell^2} - x_k)^\top a_i a_i^\top (x^\star_{\ell^2} - x_k)$, we have $\mathcal{L}(x_k) = \sum_{i=1}^m (x^\star_{\ell^2} - x_k)^\top a_i a_i^\top (x^\star_{\ell^2} - x_k) \geq \mu \left\| x^\star_{\ell^2} - x_k \right\|^2$. Consequently, $\left\| x^\star_{\ell^2} - x_k \right\|^2 \to 0$ almost surely, as desired. □

Although in most chapters in the manuscript we either focus only on optimization guarantees (proving that we converge to a minimizer of the empirical risk or generalization error) or on generalization guarantees of some given estimators, one needs to keep in mind that both are intertwined, via implicit regularization phenomena. This is particularly the case in modern machine learning applications, where models are largely overparametrised and infinitely many perfect interpolators exist, making it difficult to know which ones will generalize well. The distributed algorithms we consider next are first-order optimization algortihms, and satisfy black-box first-order assumptions as defined in [Bub15, Section 3.5]. The analysis provided just above directly extends to this whole class of algorithms and thus to algorithms developed in latter chapters.

However, quite importantly, it is not always the case that all first-order algorithms, whatever their hyperparameters (here the stepsizes and batchsizes), converge to the same solution. It has frequently been observed that in modern deep learning, parameter-tuning plays not only plays a crucial role in convergence properties on the training set, but also on the generalization guarantees of the retrieved solution: both are intertwined.

In Chapter 11, we study the implicit regularization of stochastic gradient descent for simple non-convex models (diagonal linear networks). We highlight that the hyperparameters of the algorithm (stepsize and batchsize) lead to SGD converging to interpolating solutions that may vary a lot, which is not the case for the very simple convex model (quadratic linear regression) considered in the previous example.

1.2. Distributed learning setting

Now that we have introduced the basis of ML theory we require, we are armed to introduce the distributed learning setting in this section, before highlighting the settings we will consider in this thesis in latter sections. In distributed learning, the purely sequential nature of first-order methods such as (GD) or (SGD) above changes. Indeed, we now assume that there are n computing nodes, that may each compute information in parallel. However, depending on the assumptions we make — the existence of a central orchestrator, compute nodes that possess private data, restricted communications, ... —, the nature of the resulting algorithms may drastically change.

The common ground of the different flavors of distributed learning we will consider, is that the problem at hand can be written as follows. The n compute nodes, indexed by $i \in [n]$, each possess a local function f_i that may be accessed only at node i. This function f_i can either be the empirical risk associated to the data held by node i, or directly its generalization

Algorithm 1.1: DISTRIBUTED SGD

Input: Initialization x_0,
stepsizes $\{\eta_k\}_{k\geq 0}$,
number of epochs K

for $k = 0$ *to* $K-1$ **do**
  Server sends x_k to all nodes i
  for *all nodes* i *in parallel* **do**
    Compute $g_{k,i}$ unbiased estimate of $\nabla f_i(x_k)$
    Send it back to the server
  Server updates $x_{k+1} = x_k - \frac{\eta_k}{n}\sum_{i=1}^n g_{k,i}$

Output: x_K

Algorithm 1.2: LOCAL SGD

Input: Initialization x_0,
stepsizes $\{\eta_{k,h}\}_{k,h\geq 0}$,
epochs K, local steps H

for $k = 0$ *to* $K-1$ **do**
  i receives $y_{0,i}^k = x_k$ from server
  for *all nodes* i *in parallel* **do**
    for $h = 0$ *to* $H-1$ **do**
      Compute $g_{h,i}^k$ and:
      $y_{h+1,i}^k = y_{h,i}^k - \eta_{k,h} g_{h,i}^k$
    Send $x_{k+1,i} = y_{H,i}^k$ to server
  $x_{k+1} = \frac{1}{n}\sum_{i=1}^n x_{k+1,i}$

Output: x_K

Figure 1.1 – *Distributed algorithms in the centralized setting: Distributed (or Minibatch) SGD and Federated Averaging (or Local SGD)*

error or expected risk with respect to some local data distribution. Correspondingly,

$$f_i(x) = \frac{1}{m}\sum_{j=1}^m F_i(x,\xi_{ij})\,, \quad \text{or} \quad f_i(x) = \mathbb{E}\left[F_i(x,\xi)\right] \quad \text{where} \quad \xi \sim \mathcal{D}_i\,,$$

for $\mathcal{D}_i$ the local distribution of agent i in the latter and $\xi_{i1},\dots,\xi_{im}$ its local data samples in the former case. The function F_i corresponds to $F_i(x,\xi) = \ell(\varphi(x,a),b)$ for data sample $\xi = (a,b)$ in the notations of the previous section.

The goal of distributed learning is then to minimize the averaged function f:

$$\min_{x\in\mathbb{R}^d}\left\{f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x)\right\}. \tag{1.5}$$

In the case where local functions f_i are empirical risks, this amounts to minimizing the empirical risk over all samples $\{\xi_{ij}\}$, while if the local functions are local generalization errors with respect to some local distributions $\mathcal{D}_i$, this amounts to minimizing the generalization error with respect to the mixture of all local distributions $\mathcal{D}_i$. In both cases, a single model is learnt for all compute nodes, and functions f_i can either be heterogeneous or all equal.

Depending on the scenarios, compute nodes may also be referred to as machines, agents or users.

1.2.1. Central-server (centralized) setting: distributing SGD over different compute nodes

The most basic communication setting is the central-server (or *centralized*) one: there exists a central unit that aggregates information from all compute nodes, perform computations, and communicate with the compute nodes. This is summarized in Figure 1.2 (left): agents (bottom line) communicate with the server through the communication links. In this subsection, we present two very classical algorithmic approaches to minimize the objective defined in Equation (1.5) in a distributed way in the centralized setting, in order to introduce the decentralized setting in next subsection and to better understand the settings we will introduce later in this chapter.

Distributed SGD (or Minibatch SGD). In the centralized setting, the simplest algorithm consists in parallelizing the computations of gradients. At each iteration, *(i)* the server sends the current model x_k to all n agents; *(ii)* for all i, agent i computes $g_{k,i}$ a stochastic gradient (an unbiased estimate of $\nabla f_i(x_k)$) and sends it back to the server; *(iii)* the server updates the model by aggregating all stochastic gradients. Note that $\frac{1}{n}\sum_{i=1}^n g_{k,i}$ is then an unbiased stochastic gradient of $f = \frac{1}{n}\sum_i f_i$. This is summarized in Algorithm 1.1.

Distributed SGD has two advantages: the first one is computational, since all stochastic gradients computed at each epoch are computed in parallel. The second advantage is that the *variance* of the stochastic gradient $\frac{1}{n}\sum_{i=1}^n g_{k,i}$ used to update x_k is decreased by n the number of nodes, compared to the variance of each stochastic gradients. We thus expect the convergence of this algorithm to require less epochs than simply using one stochastic gradient (that will have a higher variance) at a time: since computes are paralleled, if all machines compute at the same speed, the time per iteration will be the same, while each iteration of distributed SGD will be more efficient. If σ^2 is an upper bound on the variance of stochastic gradients $g_{k,i}$ and on the population variance (*i.e.*, $\mathbb{E}\left[\|g_{k,i} - \nabla f_i(x_k)\|^2 | x_k\right] \leq \sigma^2$ and $\frac{1}{n}\mathbb{E}\left[\|\nabla f_i(x_k) - \nabla f(x_k)\|^2 | x_k\right] \leq \sigma^2$), and local functions f_i are convex and $L-$smooth and $B^2 \geq \|x_0 - x^\star\|^2$ where $x^\star$ minimizes f, then the expected rate of convergence will be [GG23]:

$$\mathbb{E}\left[f(x_k) - f(x^\star)\right] = \mathcal{O}\left(\frac{LB^2}{K} + \sqrt{\frac{\sigma^2 B^2}{nK}}\right). \tag{1.6}$$

This can be obtained by directly adapting Proposition 1.1.2. We thus observe a speedup proportional to n the number of compute nodes in the lower order term (the second term) compared to vanilla SGD as in Equation (1.1), while each iteration is not expected to take much longer than SGD that uses only one stochastic gradient $g_{k,i}$ at each epoch. In the case where compute nodes hold local functions f_i that are heterogeneous and that for instance correspond to empirical risks computed with some local private data of local node i, this algorithm is naturally distributed. In the case where all data samples are held at the server, the server itself sends data samples to compute nodes at each iteration to compute stochastic gradients in parallel: in this configuration, this algorithm is in fact simply a distributed version of *Minibatch SGD*, has a long history [ZWLS10, CSSS11, DGBSX12] and is commonly used in practice [GDG+17].

Local SGD (or Federated Averaging). The previous algorithm however requires that for each stochastic gradient step, the server communicates with all nodes. This can become quite computationally inefficient if communications are more expensive than computations. Hence the idea behind the Local SGD algorithm (or equivalently, Federated Averaging, depending on the terminology used) summarized in Algorithm 1.2: instead of simply performing a single stochastic gradient step for each communication, compute nodes perform $H \geq 2$ local stochastic gradient steps [MMR+17, WPS20b, Sti18]. This algorithm benefits from an increased efficiency per iteration. Under similar assumptions as above, we expect the following rate [KLB+20]:

$$\mathbb{E}\left[f(x_k) - f(x^\star)\right] = \mathcal{O}\left(\frac{LB^2}{K} + \left[\frac{\sqrt{L}\sigma B^2}{K}\right]^{2/3} + \sqrt{\frac{\sigma^2 B^2}{nHK}}\right). \tag{1.7}$$

The lower order term (the last one) thus benefits from a speedup proportional to H the number of local steps compared to Minibatch/distributed SGD (Equation (1.6)) while requiring the same number of communications.

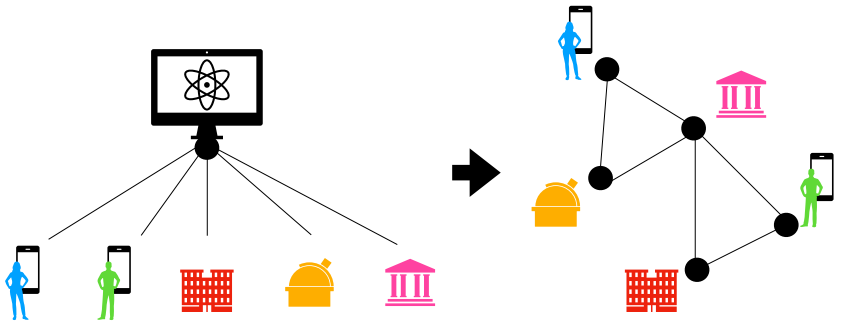


Figure 1.2 – *Centralized to decentralized setting*

The centralized setting however has one major drawback: all communications happen between compute nodes and the server. Consequently, if the server fails, the whole procedure stops. This will happen if the scale of the problem and the number of compute nodes increase too much: since the server has a limited bandwidth (the rate of data transfer), if the number of compute nodes become too large, then it will not be able to handle all communications and will thus stop or slow down the procedures. Indeed, workers will need to wait for the server to be available, thus loosing the benefits of parallelization.

1.2.2. From centralized distributed learning to decentralized learning and gossip algorithms

The decentralized setting gets rids of a central server, as illustrated in Figure 1.2: compute nodes communicate with each other in a peer-to-peer fashion. Formally, compute nodes are assumed to be nodes of an undirected graph $G = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = [n]$ and the edges $\mathcal{E}$ correspond to pairwise communications that are possible only between adjacent nodes in the graph. In the centralized setting, the server kept a copy x_k of the current model and broadcasted it to compute nodes: this is no longer possible in the decentralized setting. Thus, compute node that corresponds to node $v \in \mathcal{V}$ in the graph keeps a local copy x_v^k of the current iterate .[1]

The two key ingredients behind centralized distributed optimization algorithms (such as Distributed Minibatch SGD or Local SGD, Figure 1.1) are *(i)* local gradient computations at the location of compute nodes and eventually local gradient steps; *(ii)* communication with the server, that averages the values computed by the nodes. While *(i)* does not suffer from decentralization, *(ii)* can no longer be done in the decentralized setting. Thus, communications with the server that abstractly consisted of *global* averagings, now need to be replaced by *local* averagings. Since communications can only happen between adjacent nodes in the graph, these averagings will be made between adjacent nodes. These local communications are formalized with *gossip matrices*.

Definition 1.2.1 (Gossip matrix)**.** *A gossip matrix on graph $G = (\mathcal{V}, \mathcal{E})$ is a symmetric matrix $W \in \mathbb{R}^{\mathcal{V}\times\mathcal{V}}$ such that the followings hold.*

1. *For all $v, w \in \mathcal{V}$, $W_{\{v,w\}} \geq 0$;*
2. *For all $v, w \in \mathcal{V}$, $W_{\{v,w\}} > 0 \implies \{v, w\} \in \mathcal{E}$;*

[1] throughout this manuscript, we prefer the notation $v \in \mathcal{V}$ instead of $i \in [n]$ when dealing with nodes in a communication graph, to highlight the graph dependency. We may however use both notations in some cases.

3. *For all* $v \in \mathcal{V}$, *we have* $\sum_{w \in \mathcal{V}} W_{\{v,w\}} = 1$.

A gossip matrix is thus simply a symmetric and *bistochastic* matrix. Given the communication graph G, gossip matrices can be easily exhibited, such as $W = I - \frac{D-A}{d}$ where d is the maximum degree of a node in the graph, D is the diagonal matrix such that $D_{v,v}$ is the degree of node v, and A is the adjacency matrix of graph G ($A_{\{v,w\}} = 1$ iff $\{v,w\} \in \mathcal{E}$).

The decentralized averaging problem. The decentralized averaging problem consists of computing, in a decentralized way (using only communications along edges of graph G), the mean of values $\left\{x_v \in \mathbb{R}^d, v \in \mathcal{V}\right\}$ held by nodes of the graph. The gossip algorithm is the most natural algorithm to solve this problem. Given a gossip matrix W, initializing the procedure with $x_v^0 = x_v$ at node $v \in \mathcal{V}$, iterates of the gossip algorithm are obtained by iterating the following recursion:

$$\forall k \geq 0\,, \quad \forall v \in \mathcal{V}\,, \quad x_v^{k+1} = \sum_{w \in \mathcal{N}_v} W_{\{v,w\}} x_w^k\,, \tag{1.8}$$

where $\mathcal{N}_v = \{w \in \mathcal{V} \,|\, \{v,w\} \in \mathcal{E}\}$ are the neighbors of node v in the graph. Due to Points 2. and 3. in Definition 1.2.1, we have $\sum_{w \in \mathcal{N}_v} W_{\{v,w\}} = 1$: x_v^{k+1} is obtained by performing a weighted average of all the values x_w^k for w adjacent to v in the graph. Hence, Equation (1.8) corresponds exactly to a local averaging, that consists of an approximation of the global averaging in the decentralized setting.

Equation (1.8) can be written more compactly in matrix form. Denoting $X = (x_v)_{v \in \mathcal{V}} \in \mathbb{R}^{\mathcal{V} \times d}$ and $X^k = (x_v^k)_{v \in \mathcal{V}} \in \mathbb{R}^{\mathcal{V} \times d}$ for $k \geq 0$, we simply have:

$$\forall k \geq 0\,, \quad X^{k+1} = W X^k\,. \tag{1.9}$$

This sequence satisfies the following property.

Proposition 1.2.1. *Let* $\bar{x} = \frac{1}{n} \sum_{v \in \mathcal{V}} x_v$ *the mean to be approximated, and let* $\rho = \min(1 - \lambda_2, 1 + \lambda_n)$, *where* $\lambda_1 \geq \ldots \lambda_n$ *are the eigenvalues of* W. *Then, for all* $k \geq 0$,

$$\frac{1}{n} \sum_{v \in \mathcal{V}} \left\| x_v^k - \bar{x} \right\|^2 \leq \frac{(1-\rho)^{2k}}{n} \sum_{v \in \mathcal{V}} \|x_v - \bar{x}\|^2\,.$$

Proof. Let $\mathbb{1} \in \mathbb{R}^{\mathcal{V}}$ be the vector with all entries equal to 1 and $\bar{X} = \mathbb{1}\bar{x}^\top = (\bar{x}|\ldots|\bar{x})^\top \in \mathbb{R}^{\mathcal{V} \times d}$. We have $W\mathbb{1} = \mathbb{1}$, so that $W\bar{X} = \bar{X}$, leading to $X^{k+1} - \bar{X} = WX^k - \bar{X} = W(X^k - \bar{X})$. Then, $\sum_{v \in \mathcal{V}} x_v^{k+1} = \sum_{v \in \mathcal{V}} x_v^k$ since multiplying by W preserves the mean, leading to $\sum_{v \in \mathcal{V}} x_v^k = n\bar{x} = \sum_{v \in \mathcal{V}} x_v$ by induction.

Since W is a symmetric bistochastic matrix, using the Perron-Frobenius theorem, its spectrum consists of eigenvalues $1 = \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq -1$, where $\lambda_1 = 1$ is associated to the eigenvector $\mathbb{1}$. Since $X^k - \bar{X}$ is orthogonal to $\mathbb{1}$, we have $\left\|W(X^k - \bar{X})\right\| \leq \max_{k \geq 2} |\lambda_k| \left\|X^k - \bar{X}\right\| = (1-\rho)\left\|x^k - \bar{X}\right\|$, by definition of ρ. This leads to the desired result by induction. □

Hence, the iterates of the gossip algorithm defined in Equation (1.8) converge to the desired mean, as long as $\rho > 0$ and $k \gg \rho^{-1}$. For the gossip matrix $W = I + \frac{D-A}{d}$ described previously, as long as the graph G is connected, $\rho > 0$ is satisfied. Note that for instance for the complete graph, this matrix boilds down to $W = \mathbb{1}\mathbb{1}^\top/n$ (all entries equal to $1/n$), and $\rho = 1$: only one iteration is required, since this iteration consists of a global averaging.

The bottleneck of the gossip algorithm as formulated in Equation (1.8) is that at each iteration, every node needs to communicate with every neighbor: this can be quite costly and might require heavy synchronization costs. An alternative to this is to use varying

communication matrices, that may only activate a few communication edges in the graph at each round:

$$\forall k \geq 0\,, \quad \forall v \in \mathcal{V}\,, \quad x_v^{k+1} = \sum_{w \in \mathcal{N}_v} W_{\{v,w\}}^k x_w^k\,, \tag{1.10}$$

where $\left\{W^k\right\}_{k\geq 0}$ are gossip matrices. This can be written as $x^{k+1} = W^k x^k$. A classical instance of this is the *randomized gossip algorithm* proposed by [BGPS06], where $W^k = W_{\{v_k,w_k\}}$ for

$$W_{\{v_k,w_k\}} = I - \frac{(e_{v_k} - e_{w_k})(e_{v_k} - e_{w_k})^\top}{2}\,,$$

where (e_v) is the canonical basis of $\mathbb{R}^{\mathcal{V}}$. In this case, the iteration in Equation (1.10) reads as

$$\begin{aligned} x_v^{k+1} &= x_v^k \quad \text{for} \quad v \notin \{v_k, w_k\}\,, \\ x_{v_k}^{k+1} &= x_{w_k}^{k+1} = \frac{x_{v_k}^k + x_{w_k}^k}{2}\,. \end{aligned}$$

This is a local averaging alongside edge $\{v_k, w_k\}$. For gossip with varying matrices, we have the following proposition, that holds under some randomness assumption on matrices W_k, that is satisfied for randomized gossip if $\{v_k, w_k\}$ is sampled uniformly at random in $\mathcal{E}$.

Proposition 1.2.2. *Assume that matrices W_k are random and independent and satisfy the following mixing property: there exists some $\rho' > 0$ such that for all $k \geq 0$ and all $Y \in \mathbb{R}^{\mathcal{V}\times d}$, we have $\mathbb{E}\left[\left\|W^k(Y - \bar{Y})\right\|^2\right] \leq (1-\rho')\left\|Y - \bar{Y}\right\|^2$ where $\bar{Y} = \frac{\mathbb{1}\mathbb{1}^\top}{n} y$.*[2] *Then,*

$$\frac{1}{n}\sum_{v\in\mathcal{V}} \left\|x_v^k - \bar{x}\right\|^2 \leq \frac{(1-\rho')^k}{n} \sum_{v\in\mathcal{V}} \|x_v - \bar{x}\|^2\,.$$

Proof. As in previous proposition, we have $\sum_v x_v^k = n\bar{x}$ for all k, so that $\bar{X} = \frac{\mathbb{1}\mathbb{1}^\top}{n} X^k$, leading to $\mathbb{E}\left[\left\|W^k(X^k - \bar{X})\right\|^2 | X^k\right] \leq (1-\rho')\left\|X_k - \bar{X}\right\|^2$, since W^k is independent from previous matrices $W^\ell, \ell < k$ and thus from x^k. We then conclude by induction. □

We can further explicit what ρ' will look like for randomized gossip. If $\{v_k, w_k\}$ is chosen at random independently from the past with probability $p_{\{v_k,w_k\}}$ (such that $\sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}} = 1$), then for all $Y \in \mathbb{R}^{\mathcal{V}}$ such that $\langle \mathbb{1}, Y\rangle = 0$, we have , since $W_{\{v_k,w_k\}}$ is an orthogonal projection:

$$\begin{aligned} \mathbb{E}\left[\left\|W_{\{v_k,w_k\}} Y\right\|^2\right] &= \sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}} Y^\top W_{\{v_k,w_k\}}^\top W_{\{v_k,w_k\}} Y \\ &= \sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}} Y^\top W_{\{v_k,w_k\}} Y \\ &= \sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}} Y^\top \left(I - \frac{(e_v - e_w)(e_v - e_w)^\top}{2}\right) Y \\ &= \sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}} Y^\top Y - \sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}} Y^\top \frac{(e_v - e_w)(e_v - e_w)^\top}{2} Y \\ &= \|Y\|^2 - \sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}} (y_v - y_w)^2 \end{aligned}$$

[2] if matrices W_k are deterministic, this boils down to assume that all W_k satisfy the assumption of Proposition 1.2.1 with $\rho' \sim 2\rho$.

Table 1.1 – *Constants ρ and ρ' for various simple graphs for respectively constant gossip matrix $W = I - \frac{D-A}{d}$ for ρ in Proposition 1.2.1 and randomized gossip matrices with edges sampled uniformly at random for ρ' in Proposition 1.2.2. n is the number of nodes in the graph. In the second half of the table, we show the* efficiency *of each method, that refers to the number of communications performed per epoch, times ρ^{-1} or ρ'^{-1} for respectively gossip with $W = I - \frac{D-A}{d}$ (1) and for randomized gossip (2).*

Graph	Expander	D-dim. Torus	Complete	Line	Ring
ρ ((1), Prop 1.2.1)	$\mathcal{O}(1)$	$\mathcal{O}\Big(\frac{1}{n^{\frac{2}{D}}}\Big)$	1	$\mathcal{O}\big(\frac{1}{n^2}\big)$	$\mathcal{O}\big(\frac{1}{n^2}\big)$
ρ' ((2), Prop 1.2.2)	$\mathcal{O}\big(\frac{1}{n}\big)$	$\mathcal{O}\Big(\frac{1}{n^{1+\frac{2}{D}}}\Big)$	$\mathcal{O}\big(\frac{1}{n}\big)$	$\mathcal{O}\big(\frac{1}{n^3}\big)$	$\mathcal{O}\big(\frac{1}{n^3}\big)$
Efficiency (1)	$\mathcal{O}(n)$	$\mathcal{O}\Big(Dn^{1+\frac{2}{D}}\Big)$	n^2	$\mathcal{O}\big(n^3\big)$	$\mathcal{O}\big(n^3\big)$
Efficiency (2)	$\mathcal{O}(n)$	$\mathcal{O}\Big(n^{1+\frac{2}{D}}\Big)$	$\mathcal{O}(n)$	$\mathcal{O}\big(n^3\big)$	$\mathcal{O}\big(n^3\big)$

$$= \|Y\|^2 - Y^\top \Delta(p) Y \,,$$

where the matrix $\Delta(p)$ is such that $Y^\top \Delta(p) Y = \sum_{\{v,w\} \in \mathcal{E}} p_{\{v,w\}} (y_v - y_w)^2$. Then, ρ' in Proposition 1.2.2 is exactly the smallest eigenvalue of $\Delta(p)$ on the orthogonal of $\mathbb{1}$. We will see below that $\Delta(p)$ is in fact a graph Laplacian.

From Table 1.1, we thus see that if each iteration of plain gossip ($W = I - \frac{D-A}{d}$) or randomized gossip takes the same amount of time, plain gossip will be fastest, since ρ' is smaller than ρ, with a multiplicative factor $1/n$ that can be large. However, comparing the efficiency of each communication, randomized gossip appears to be better. Overall, the use of plain or randomized gossip depend on the constraints of the problem and the level of synchronicity allowed.

From gossip algorithms to decentralized optimization. From gossip algorithms that aim at approximating the mean operation performed by the central server, how can we derive decentralized learning algorithms? First, note that the iterations of Minibatch SGD in Algorithm 1.1, that write $x_{k+1} = x_k - \frac{\eta_k}{n} \sum_{i=1}^n g_{k,i}$, are equivalent to:

$$X^{k+1} = W\big(X^k - \eta^k G^k\big) \,,$$

where $W = \frac{\mathbb{1}\mathbb{1}^\top}{n}$, $X^0 \in \mathbb{R}^{n \times d}$ is initialized as $X_i^0 = x_0$, and $G^k \in \mathbb{R}^{n \times d}$ is the concatenated matrix of all stochastic gradients ($G_i^k = g_{k,i}$). Then, we have that $X_i^k = x_k$ for all $i \in [n]$.

Similarly, the iterates of Local SGD (Algorithm 1.2) can be written as:

$$X^{r+1} = W_r\big(X^r - \eta_r G^r\big) \,,$$

where for $r+1$ multiple of H we have $W_r = \frac{\mathbb{1}\mathbb{1}^\top}{n}$ and else we have $W_r = I$. Then, for all $r+1$ multiple of H, $X_i^{r+1} = x_{\frac{r+1}{H}}$. The two centralized distributed algorithms we presented can thus be naturally cast in a decentralized way (without any need of quantity shared by the server), involving the gossip matrices $W = \frac{\mathbb{1}\mathbb{1}^\top}{n}$ (full communications on the complete graph) and $W = I$ (no communications at all). These algorithms then correspond to alternating between communications using gossip matrices and stochastic gradient steps on models held at node locations.

This leads to the following natural idea for a decentralized version of distributed SGD,

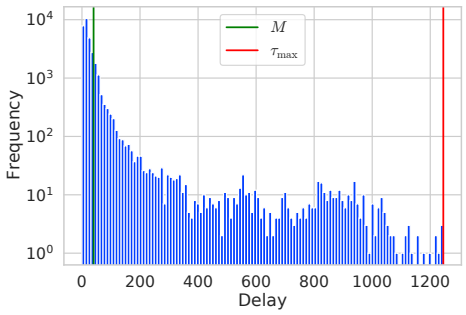


Figure 1.3 – *Distributions of delays in a centralized setting using $M = 48$ compute nodes [MBEW22] for asynchronous SGD (Algorithm 1.3).*

called *Decentralized SGD*. Nodes locally hold models x_v^k, that are iteratively updated as:

$$x_v^{k+1} = \sum_{w \in \mathcal{N}_v} W_{\{v,w\}}^k \left(x_w^k - g_{w,k} \right), \tag{1.11}$$

where $g_{w,k}$ is an unbiased estimate of $\nabla f_w(x_w^k)$, and W^k a gossip matrix. Denoting $X^k, G^k \in \mathbb{R}^{\mathcal{V} \times d}$ the concatenated matrices with $X_v^k = x_v^k$ and $G_v^k = g_{v,k}$, the iterates of Decentralized SGD (Equation (1.11)) write as:

$$X^{k+1} = W_k \left(X^k - \eta_k G^k \right),$$

which is a direct generalization of the above centralized algorithms to the decentralized setting.

Decentralized SGD as in Equation (1.11) or variants of these iterates have been studied in several previous works such as [KLB+20, LZZ+17, LHLL15, NOR18]. Overall, decentralized SGD benefits from paralleled computations (all nodes can compute stochastic gradients in parallel), and communications do not suffer from a single bottleneck as in the centralized setting. Under similar assumptions on local functions and stochastic gradients than in Equation (1.6) and if gossip matrices satisfy the assumptions of Proposition 1.2.2, Decentralized SGD satisfies [KLB+20]:

$$\mathbb{E}\left[f(\bar{x}^K) - f(x^\star) \right] = \mathcal{O}\left(\frac{LB^2}{\rho' K} + \left[\frac{\sqrt{L}\sigma}{\rho' K} \right]^{2/3} + \sqrt{\frac{B^2\sigma^2}{nK}} \right),$$

for $\bar{x}^K = \frac{1}{nK} \sum_{v \in \mathcal{V}, k < K} x_v^k$, where the graph dependency appears through parameter ρ': the more the graph is connected, the larger ρ' is.

There exist many other decentralized optimization algorithms, that all possess their pros and cons. We only presented here the simplest version, Decentralized SGD, for its simplicity, in order to highlight its link with the gossip averaging problem and its graph dependency through ρ'. We refer the interested reader to the survey [GRB+22] on recent advances in decentralized optimization.

Now that we have introduced the basics of ML theory and distributed optimization we require, we can introduce the settings we consider in this manuscript.

1.3. Asynchronous optimization

Algorithm 1.3: Asynchronous SGD

Input: initialization $x_0 \in \mathbb{R}^d$ at server, stepsizes $\eta_k > 0$
Each compute node $i \in [n]$ begins calculating $g_{i,0}$ unbiased stochastic gradient of $\nabla f_i(x_0)$;
for $k = 1, 2, \ldots$ **do**
    Stochastic gradient $g_{i_k,k-\tau(k)}$ — unbiased estimate of $\nabla f_{i_k}(x_{k-\tau(k)})$ arrives from some worker i_k
    Update: $x_k = x_{k-1} - \eta_k g_{i_k,k-\tau(k)}$
    Send $\mathbf{x}_k$ to node i_k, which begins calculating some unbiased estimate of $\nabla f_{i_k}(x_{k+1})$
end for

1.3.1. Asynchronous learning in the centralized setting

From what we saw in last section, there are many ways to parallel the minimization of Equation (1.5) in distributed optimization, both in the centralized or the decentralized setting. Taking for instance the simplest algorithm — distributed minibatch SGD, Algorithm 1.1 —, we observe that if the n compute nodes are *heterogeneous* in terms of computational speed, then the whole procedure will work at the speed of the slowest node. In Figure 1.3, we see that in cluster environment, the delays incurred by directly updating the model upon reception of a stochastic gradient are highly heterogeneous: working at the speed of the slowest node may thus, in some cases, lead to loose all the benefits of parallelization.

For instance, if $n/2$ nodes compute stochastic gradients in s_- seconds, while the second half compute stochastic gradients in s_+ seconds, each iteration of Algorithm 1.1 will take s_+ seconds. If $s_+ \gg s_-$, we would be better off by discarding this half of the compute nodes. The same phenomenon would happen for Local SGD (Algorithm 1.2). More generally, if node i computes stochastic gradients in s_i seconds, given a time budget of S seconds, Algorithm 1.1 will be able to perform a number of iterations of:

$$K_{\text{synch}} = \frac{S}{s_{\max}},$$

where $s_{\max} = \max_{i \in [n]}$ is the maximal latency. According to Equation (1.6), the time (in seconds) required to reach a precision ε would then be proportional to

$$S_{\text{sync}}(\varepsilon) = s_{\max} \max(c_1 \varepsilon^{-1}, c_2 n^{-1} \varepsilon^{-2}) \quad \text{seconds},$$

where c_1, c_2 are constants related to the regularity of the optimization functions and the noise of the stochastic gradients. This expression is directly proportional to the maximum latency amongst compute nodes. We clearly see here that K_{synch} is not robust to *stragglers*: compute nodes that are slower than the others.

The role of asynchronous optimization is thus to derive algorithms and guarantees that are robust to these stragglers: the performances should not be directly impacted by a few slow workers. The asynchronous counterpart to Algorithm 1.1 is *Asynchronous SGD*, presented in Algorithm 1.3. Every node compute its stochastic gradients at its own speed, and directly sends it back to the server when computes are finished. Upon reception of a stochastic gradient from a compute node, the server directly updates its model, and sends back the updated model to the worker that sent the stochastic gradient.

The advantage of Asynchronous SGD over Algorithm 1.1 is that there is no form of waiting: machines work at their own speed, whether or not there are straggler nodes. The drawback is then that in the update of the model, as seen in Algorithm 1.3, the stochastic gradient used is *delayed*: it is computed at some model $x_{k-\tau(k)}$ instead of x_k, leading to biased updates and thus decreased efficiency.

However, the existing theoretical guarantees for Asynchronous SGD are disappointing, and the typical approach to analyzing the algorithm involves assuming that all of the delays are either the same, $\tau(k) = \tau$, or at least upper bounded, $\tau(k) \leq \tau_{\max}$ [AD11, MPP+17, LPLJ18, ASS20, SK20]. Forgetting about the statistical rate (the ε^{-2} dependency in the number of iterations required), these analyses then show that the number of updates needed to reach accuracy ε is of order $\varepsilon^{-1}\tau_{\max}$. In a time window of S seconds, Asynchronous SGD performs on average $\sum_{i=1}^n \frac{S}{s_i}$ iterations since every iteration takes on average a time $\left(\sum_{i=1}^n \frac{1}{s_i}\right)^{-1}$. $\tau_{\max}$ is then of order $\sum_{i=1}^n \frac{s_{\max}}{s_i}$, so that previous analyses suggest that asynchronous SGD takes a time proportional to:

$$S_{\text{asynch}}(\varepsilon) = \varepsilon^{-1} \sum_{i=1}^n \frac{s_{\max}}{s_i} \left(\sum_{i=1}^n \frac{1}{s_i}\right)^{-1} = \varepsilon^{-1} s_{\max}\,,$$

to reach a precision ε. This leads to no speedup compared to the synchronous algorithms!

This quite unfortunate, and goes against the intuition that asynchronous algorithms should be faster than their synchronous counterparts. Specifically, suppose we have two parallel workers—one fast device that needs just 1μs to calculate a stochastic gradient, and one slow device that needs 1s. If we use these two machines to implement Asynchronous SGD, the delay of the slow device's gradients will be 1 million, because in the 1 second that we wait for the slow machine, the fast one will produce 1 million updates. Consequently, the analysis based on $\tau_{\max}$ degrades by a factor of 1 million. But on further reflection, Asynchronous SGD should actually do very well in this scenario, after all, 99.9999% of the SGD steps taken have gradients with no delay! Even if one update in a million has an enormous delay, it seems fairly clear that a few badly out-of-date gradients should not be enough to ruin the performance of SGD—a famously robust algorithm.

Contributions of Chapter 3. A key challenge in asynchronous optimization is thus to provide provable *asynchronous speedups*, *i.e.*, guaranteeing that $S_{\text{asynchronous}}(\varepsilon) \ll S_{\text{synchronous}}(\varepsilon)$ in the presence of stragglers. This is exactly the purpose of Chapter 3, where we prove that $\frac{S_{\text{synch}}(\varepsilon)}{S_{\text{asynch}}(\varepsilon)} = \frac{1}{n}\sum_{i=1}^n \frac{s_{\max}}{s_i} = \frac{s_{\max}}{\bar{s}} \geq 1$, where $\bar{s}$ is the harmonic mean of the s_i's, and may be much smaller than $s_{\max}$ in the presence of stragglers. As we will see below, the story in decentralized communications becomes however more complicated.

1.3.2. Asynchronous and decentralized communications

For decentralized optimization, there are both computation and communication latencies. Computation ones should not be too different than from the centralized case (this is the purpose of Chapter 5), and the interesting part of the story comes with communicationz and delays in the gossip part of Decentralized SGD. The question then is: how can the gossip averaging algorithm be made asynchronous and what would be its asynchronous speedup ?

First, as in the centralized setting, let $s_{\{v,w\}}$ be the time a communication takes between adjacent nodes v and w, and let $s_{\max} = \max_{\{v,w\}\in\mathcal{E}} s_{\{v,w\}}$. For the information to flow from a node v to another node w in the graph, a time $\text{dist}_s(v,w) = \inf_{v=u_0\sim\ldots\sim u_p=w} \sum_{k=0}^{p-1} s_{\{u_k,u_{k+1}\}}$ is necessary: $d_s(v,w)$ is the time to reach w from v using the fastest path between them. Thus, the time required to compute the mean of decentralized values is lower bounded by:

$$\text{diam}_s = \max_{(v,w)\in\mathcal{V}^2} \text{dist}_s(v,w)\,.$$

This quantity can be referred to as the graph time-diameter. In the case where all communication times are the same and equal to τ, this quantity boilds down to $\tau \times \text{diam}$, where diam is the diameter of the graph. However, these quantities (graph diameters) are not what

decentralized algorithms are about.

If delays are all the same, then synchronous algorithms should be able to depend on τdiam only, and they do: one could consider the simple algorithm that consist of choosing any node v_0 in the graph. Then, all nodes $w \neq v_0$ send their values to this node through the shortest path between v_0 and w: this takes a time at most τdiam for all nodes, and thus the mean can be computed in a time τdiam. With heterogeneous delays, this is simply replaced by diam_s. However, these algorithms are not decentralized: they simply mimic the behavior of centralized algorithms, the server being replaced by a single compute node, therefore choosing the worst of both centralized and decentralized worlds.

We are now going to give heuristics that will highlight what the asynchronous speedup could be for decentralized mean computations through gossip-like communications. As a synchronous baseline (the equivalent of Algorithm 1.1 for our decentralized mean estimation problem), we consider the vanilla gossip iterations, with constant matrix $W = I - \frac{1}{d}(D - A)$. For $d-$regular graphs (graphs for which the degree of all nodes is d), we have $W_{v,w} = \frac{1}{d}$. Now, the graph-dependent quantity in Proposition 1.2.1 is ρ, which can in fact be related to the smallest non-null eigenvalue of the matrix $\frac{D-A}{d}$, a weighted graph Laplacian.

Definition 1.3.1 (Weighted graph Laplacian)**.** *Let $\nu = (\nu_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$ be non-negative weights. The Laplacian matrix of graph G with weights ν — referred to as weighted graph Laplacian with weights ν — is the symmetric matrix $\Delta(\nu) \in \mathbb{R}^{\mathcal{V}\times\mathcal{V}}$ such thas:*

1. *For all $v, w \in \mathcal{E}$, $\Delta(\nu)_{v,w} = 0$ if $\{v, w\} \notin \mathcal{E}$;*
2. *For all $\{v, w\} \in \mathcal{E}$ such that $v \neq w$, $\Delta(\nu)_{\{v,w\}} = -\nu_{\{v,w\}}$;*
3. *For all $v \in \mathcal{V}$, $\Delta(\nu)_{v,v} = \sum_{w\in\mathcal{N}_v\setminus\{v\}} \nu_{\{v,w\}}$.*

Weighted graph Laplacian matrices are symmetric positive semi-definite matrices and always satisfy $\mathbb{1} \in \ker(\Delta(\nu))$. If the graph is connected and weights are non-null, 0 has multiplicity 1 as an eigenvalue of $\Delta(\nu)$. The second smallest non-null eigenvalue of $\Delta(\nu)$ is thus positive: we denote it as $\lambda_2(\nu)$, and refer to it as the eigengap of the weighted graph Laplacian with weights ν. A simple computation yields that for all $Y \in \mathbb{R}^{\mathcal{V}}$, $Y^\top \Delta(\nu) Y = \frac{1}{2}\sum_{\{v,w\}\in\mathcal{E}} \nu_{\{v,w\}}(y_v - y_w)^2$.

The matrix $\frac{D-A}{d}$ is thus a weighted graph Laplacian, with constant weights equal to $1/d$: $\frac{D-A}{d} = \Delta(1/d)$, and ρ is related to $\lambda_2(1/d)$. The time per iteration of synchronous gossip being $s_{\max} = \max_{\{v,w\}\in\mathcal{E}} s_{\{v,w\}}$, the typical time to wait for convergence (in seconds) for synchronous gossip is thus $S_{\text{synch}} = \rho^{-1} s_{\max}$, which writes as:

$$S_{\text{synch}} = \frac{d}{\lambda_2\left(\frac{1}{s_{\max}}\right)},$$

since $\lambda_2(1/d)/s_{\max} = \lambda_2(1/s_{\max})/d$ by linearity of the Laplacian in its weights.

Now, what would be a good first candidate for asynchronous gossip? Historically, asynchronous gossip referred to randomized gossip [BGPS06] previously introduced, where pairwise communications happenned at the times of *Poisson point processes* The asynchronous model for gossip averagin in [BGPS06] is the following. Each edge $\{v, w\} \in \mathcal{E}$ has a clock that ticks at a *Poisson rate* of intensity $q_{\{v,w\}}$ [3]. Intensities $q_{\{v,w\}}$ are frequencies: they are homogeneous to the inverse of time. Then, at the ticking of the clock of edge $\{v, w\}$, the pairwise averaging with matrix $W_{\{v,w\}}$ is performed.

Denoting as $\mathcal{T}_{\{v,w\}} = \left\{0 < T^1_{\{v,w\}} < \ldots < T^\ell_{\{v,w\}} < \ldots\right\}$ the clock ticking times of edge $\{v, w\}$'s clock, and $\mathcal{T} = \bigcup_{\{v,w\}\in\mathcal{E}} \mathcal{T}_{\{v,w\}} = \{0 = T_0 < T_1 < \ldots < T_k < \ldots\}$ all the clock tickings, we know that the clock tickings of $\mathcal{T}$ happen at a Poisson rate $\mathcal{I} = \sum_{\{v,w\}\in\mathcal{E}} q_{\{v,w\}}$, a

[3] a Poisson clock of intensity q can be thought as, for now, a clock that has a probability $q\mathrm{d}t$ of ticking in every interval $[t, t + \mathrm{d}t)$, independently of the past, for $\mathrm{d}t$ an infinitesimal time increment.

quantity referred to as the global Poisson intensity. Now, for every global clock ticking of $\mathcal{T}$, the probability that this corresponds to a clock ticking of edge $\{v, w\}$ is exactly $p_{\{v,w\}} = \frac{q_{\{v,w\}}}{I}$.

Writing as $(x_v(t))_{t\in\mathbb{R}_{\geq 0}}$ the variable held by node $v \in \mathcal{V}$, indexed in continuous time $t \geq 0$, and by $x_v^k = x_v(T_k+)$ (T_k+ is the right-limit of T_k, and corresponds to the limit $T_k + \mathrm{d}t$ for $\mathrm{d}t \to 0$ with $\mathrm{d}t > 0$), we have that $X^k = (x_v^k)_v \in \mathbb{R}^{\mathcal{V}\times d}$ are the iterates of randomized gossip as written in Equation (1.10), for $W^k = W_{\{v_k,w_k\}}$.

But now, what about communication times, where do they appear ? Indeed, so far we have clocks that tick at random times, but that are not related to any delays, latencies, or physical constraints. This is where we are going to rely on heuristics and strong approximations, in order to be able to compare communication times. Since the communication times between adjacent nodes v and w is $s_{\{v,w\}}$ (in seconds), then in average in the *Poisson* model, the time between two tickings of edge $\{v, w\}$ should be $s_{\{v,w\}}$. Random variables $T^{\ell+1}_{\{v,w\}} - T^{\ell}_{\{v,w\}}$ are *i.i.d.* exponential random variable of parameter $q_{\{v,w\}}$: their mean is thus $1/q_{\{v,w\}}$. In order to have this quantity equal to $s_{\{v,w\}}$, we thus need $q_{\{v,w\}} = 1/s_{\{v,w\}}$.

From the discussion after Proposition 1.2.2, we have that

$$\mathbb{E}\left[\left\|X^k - \bar{X}\right\|^2\right] \leq (1 - \lambda_2(p))^k \left\|X - \bar{X}\right\|^2,$$

leading to the following result.

Proposition 1.3.1. *For all $t \geq 0$, we have:*

$$\sum_{v\in\mathcal{V}} \|x_v(t) - \bar{x}\|^2 \leq \exp(-\lambda_2(q)t) \sum_{v\in\mathcal{V}} \|x_v - \bar{x}\|^2.$$

Proof. For $t \geq 0$, let $k(t) = \sup\{k \in \mathbb{N}, T_k < t\} = |\mathcal{T} \cap [0, T)| - 1$. $k(t)$ is a discrete Poisson random variable, of parameter It: $\mathbb{P}(k(t) = k) = e^{-It}\frac{(It)^k}{k!}$. Since $\mathbb{E}\left[\left\|X(t) - \bar{X}\right\|^2\right] = \mathbb{E}\left[\left\|X^{k(t)} - \bar{X}\right\|^2\right]$, we have:

$$\begin{aligned}
\mathbb{E}\left[\left\|X(t) - \bar{X}\right\|^2\right] &= \mathbb{E}\left[\left\|X^{k(t)} - \bar{X}\right\|^2\right] \\
&= \sum_{k\geq 0} \mathbb{P}(k(t) = k)\, \mathbb{E}\left[\left\|X^k - \bar{X}\right\|^2\right] \\
&\leq \sum_{k\geq 0} e^{-It}\frac{(It)^k}{k!}(1 - \lambda_2(p))^k \left\|X - \bar{X}\right\|^2 \\
&= \sum_{k\geq 0} e^{-It}\frac{(It(1 - \lambda_2(p)))^k}{k!} \left\|X - \bar{X}\right\|^2 \\
&= e^{-It}e^{It(1-\lambda_2(p))} \left\|X - \bar{X}\right\|^2 \\
&= e^{-It\lambda_2(p)} \left\|X - \bar{X}\right\|^2
\end{aligned}$$

leading to the desired result since $Ip = q$. □

Thus, the typical time to wait before convergence is $S_{\text{asynch}} = \lambda_2(p)^{-1}$, that writes as:

$$S_{\text{asynch}} = \frac{1}{\lambda_2\left(\frac{1}{s}\right)},$$

where $\lambda_2\left(\frac{1}{s}\right)$ is the smallest non-null eigenvalue of the graph Laplacian with weight $\frac{1}{s_{\{v,w\}}}$ at edge $\{v, w\}$. Compared to S_{synch} we thus expect to pay the price of the inverse of the

eigengap of a weighted graph Laplacian too, but with weights $\frac{1}{s_{\{v,w\}}}$, instead of uniform weights $\frac{1}{s_{\max}}$. The eigengap of weighted Laplacians is a non-decreasing functions of weights: thus $\lambda_2\left(\frac{1}{s}\right) \geq \lambda_2\left(\frac{1}{s_{\max}}\right)$, leading to an asynchronous speedup!

Our heuristics did not take into account any communication delays or capacity constraints. However, this heuristics is quite insightful, as it gives us the intuition of what the asynchronous speedup will look like for asynchronous communications in decentralized algorithms: it should take the form of the eigengap of a weighted graph Laplacian, with weights that only take into account local delays, instead of worst-case delays.

Contributions of Chapters 4 and 5. In Chapter 4, we introduce *Delayed randomized gossip*: randomized gossip in continuous time as presented here, but with delays. We show that if we run randomized gossip with communication times that take $s_{\{v,w\}}$ seconds between adjacent nodes v and w, for parameters tuned accordingly, the typical time becomes, on $d-$regular graphs:

$$S_{\text{asynch}} = \frac{d}{\lambda_2\left(\frac{1}{d}\sum_{\{u,u'\}\sim\{v,w\}} \frac{1}{s_{\{u,u'\}}}\right)},$$

where the neighboring relation for edges ($\{u,u'\} \sim \{v,w\}$) means that $\{u,u'\}$ and $\{v,w\}$ share a node. Since the graph is $d-$regular, the number of elements in the sum is between $2d+1$: $\frac{1}{d}\sum_{\{u,u'\}\sim\{v,w\}} \frac{1}{s_{\{u,u'\}}}$ thus corresponds (up to a factor 2) to a local average of delays, the local average being taken in the neighborhoods of nodes v and w.

Quite surprinsingly, the same phenomenon as for Asynchronous SGD appears, but *locally* in the weights of the graph Laplacian: the dependency on $s_{\max}$ of synchronous gossip is replaced by a dependency on *local harmonic mean of local delays* in the weights of the Laplacian.

Furthermore, we extend the previous computational asynchronous speedup for Asynchronous SGD to the decentralized setting in Chapter 5.

1.3.3. Time and iteration counters in the decentralized setting

Finally, there is another huge difference between the decentralized and the centralized setting, when dealing with asynchronous communications. In both cases, time is continuous (the physical time).

In the centralized setting, events – updates and computes – can be ordered and counted as $\{T_1 < \ldots < T_k < \ldots\}$, where for instance T_k's correspond to the updates performed on the model. The integer k is then denoted as the *iteration counter*, and is a quantity known by the central server. This quantity can then be used: it is quite common that stepsizes or iterations depend on k. Two instances are for instance stepsize schedules that depend on k, the most simple ones being stepsizes $\eta_k \propto 1/\sqrt{k}$ to ensure convergence of stochastic updates. A more sophisticated instance is for instance accelerated gradient schemes and momentum, that will be of interest to us in a latter chapter (Chapter 2) when accelerating gradient descent.

However, when it comes to decentralized schemes, even though events can still be ordered and an iteration counter still exists (k in Equation (1.11)) and helps define iterates, *this iteration counter is not known by the nodes.* Indeed, a node at one end of the graph cannot know if an update is performed at the other end of the graph. Thus, except if we assume full synchronization, k cannot be used by the nodes to design stepsize schedules or accelerated methods. This difficulty for instance appears when attempting to accelerate randomized gossip algorithms "à la Nesterov": naive accelerated schemes [HBM18, CSY06, LR18] indeed require that whenever a communication between two nodes v, w adjacent in the graph ($\{v,w\} \in \mathcal{E}$) communicate together, *all* other nodes in the graph need to perform local modifications to their values (without requiring communications), or equivalently k needs to be known. While

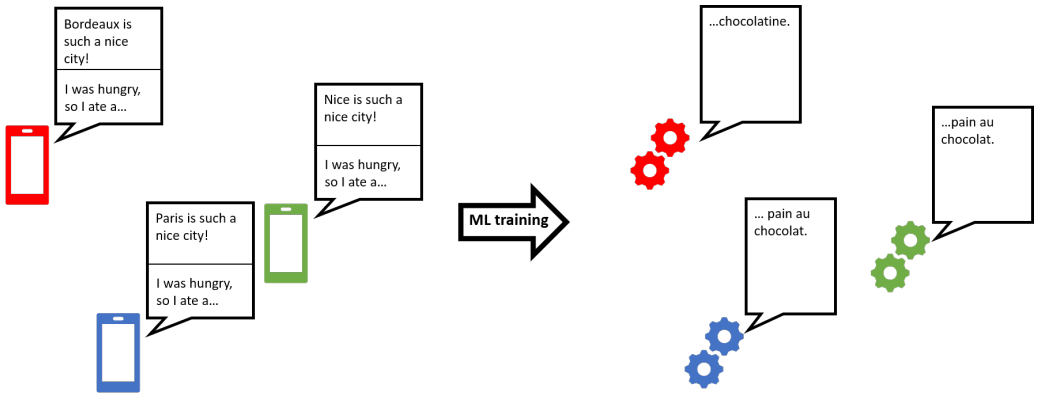


Figure 1.4 – *Next-word prediction requires some degrees of personalization.*

such procedures keep the fact that communications are pairwise as in randomized gossip, asynchrony is breached due to the high synchronicity required.

Contribution of chapter 2. In Chapter 2 we introduce an alternative to the iteration counter k: instead of using k, algorithms can build on the underlying continuous time $t \in \mathbb{R}$, that can serve as a proxy for k. For instance, stepsize schedules $\eta_k \propto 1/\sqrt{k}$ can be replaced by $\eta(t) \propto 1/\sqrt{t}$. For acceleration, the alternative to knowing k is there more sophisticated, and we elaborate further on this in Chapter 2 where we introduce the "continuized" framework to derive discrete algorithms with an underlying continuous time, and the continuized Nesterov acceleration to accelerate randomized gossip without relying on a shared iteration counter.

1.4. Personalized learning

Previous considerations aimed at accelerating the process of computing an estimator $\hat{x}$, in the form of Equation (ERM) for collaborative empirical risk minimization when samples are shared accross agents / compute nodes, or more generally $\hat{x}$ minimizer of the average of functions f_i shared amongst users (Equation (1.5)), that may be generalization errors or empirical risks. Overall, the common ground is that a *single* model $\hat{x}$ is learnt for all agents.

In the generalization bound in Equation (1.4), if each agent $i \in [n]$ has m samples, the total number of samples is nm and the generalization bound decreases with nm, instead of just m. The benefit of using all samples from all agents thus appears quite obvious: if m is too small for a single agent to have a statistical meaning — *e.g.*, think of medical tests, if the number of tests is $m = 3$, this test does not have any scientific meaning —, then an estimator based on all nm samples will have increased statistical efficiency. In the large n limit, minimizing the collaborative objective (1.5) thus leads to satistically accurate model, trained on the mixture of all distributions, whatever the scale of m (number of local samples is).

However, as always in ML problems, there are inherent limits to this. Take for instance the toy model of estimating the mean p of a Bernoulli random variable $X \sim \mathrm{Ber}(p)$, for an unknown $p \in (0,1)$, based on m *i.i.d.* samples $X_1, \dots, X_m \sim X$. We know that the minimax mean or high probability bounds for estimating p with $\hat{p}$ as a function of these m samples, is $|p - \hat{p}| = \Theta(1/m)$: one simply cannot hope to beat this bound without using further information. Thus, if agent i wishes to minimize $f_1(x) = \mathbb{E}\left[(x - X_1)^2\right]$ and thus estimate p, if for instance all other agents have objectives that are not related to estimating p, the statistical efficiency for estimating p cannot be improved (else, it would mean that agent 1 could simply generate synthetic tasks that would help, which is provably impossible [AWBR09]) by collaborating. This approach can be made more rigorous using tools such as Fano inequality.

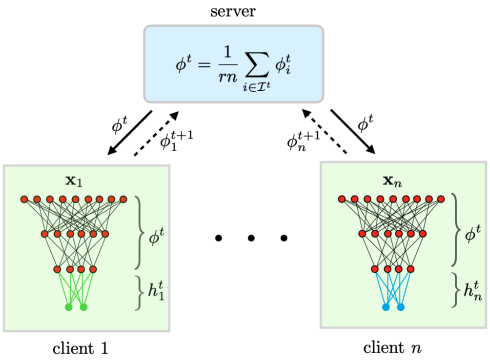


Figure 1.5 – *Shared representations in a Multi Layer Perceptron model [CHMS21]. The users share the red weights of the model, that are stored and updated at the server, while the last layers are personalized for each user.*

From a higher point of view, the fact that statistical efficieny cannot always benefit from the large pool of all users' sample can for instance be seen in next-word prediction tasks (Figure 1.4). Consider the task of predicting the next word of the sentence "I love eating ...": a model trained on all nm samples would give a single answer. However, all users might be from different backgrounds. Swiss users might be more likely to answer "gruyère", Italian users "pasta" or "pizza" or French users "chocolatine" or "baguette", not too fall too much on stereotypes (that ML models might however enforce).

The task of predicting a single model from all users, while benefiting from improved statistical efficiency on the mixture of all distributions, is thus inefficient in some unfortunate but however quite frequent cases, medical treatment predictions or other more important tasks might be some more appealing instances than the one above. In such cases, users would be better off alone than by simply training a shared model for all. But what if the number of samples m is too small? That is when a *bias-variance* tradeoff appears: the bias comes from the use of other agents' information: it is biased in the sense that this information and the samples from other agents is not sampled from the same distribution as the samples of a given agent. The more collaboration with other agents, the more bias in the computed model there is. The variance then comes from the increased statistical efficiency: the more collaboration there is, the more the number of samples the model is based on, and the less variance there is.

As we saw from the simple problem of mean estimation, assumptions are needed for collaboration to become useful at some point. There thus have been different set of assumptions, backed by empirical evidence, to design algorithms and strategies that guarantee some kind of collaboration speedup in terms of statistical guarantees. The first set of assumptions are *similarity assumptions*: each agent is assumed to be "close" to a certain number other agents. This can be formalized as a *cluster* assumption [JVB08, LSNK17, GCYR20] on the users or tasks (they form groups, and within each group/cluster the objective is the same), or as a *discrepancy-based* assumption (a metric between tasks quantitatively gives how close they are and thus how an agent can benefit from collaboration with another agent) [MMRS20, EMS22a, DW22, DKMM23]. Clusters or discrepancies are usually assumed to be unknown and based on their knowledge tasks become simpler. Under these assumptions, the objective of the problem then becomes two-fold: first, learn the structure of users *i.e.*, who is close to whom, to build some form of similarity graph of agents, and then each agents use the information of neighboring agents in this graph to compute a local model.

Assumptions can also be leveraged on the tasks themselves: they all share some common structure. The most common structural assumption that is made is the *shared representation* assumption: local datasets might be heterogeneous and in very large dimension, leading

to arbitrarily different tasks, yet high-dimensional data may share some representation in a much smaller space [BCV13]. Then, low-dimensional representations can be translated into some structure on local objectives themselves. In the case of learning deep models for instance, [CHMS21] explain that tasks may share most of the layers (Figure 1.5), while personalization can be expressed only in the last ones that are user-specific. An instance of representation assumption that is amenable for statistical analysis purposes is the *shared linear representation* assumption for linear quadratic regression: for a given task i, data is generated according to some noisy linear relation $y_{ij} = \langle w_i^\star, x_{ij} \rangle + \varepsilon_{ij}$ for data points (x_{ij}, y_{ij}), $j \in [m]$ and some $w_i^\star$ to recover. A common assumption is thus to assume that all *local heads* $w_i^\star \in \mathbb{R}^d$ for all agents lie in a small subspace of $\mathbb{R}^d$ of dimension $r \lll d$.

In both cases – user assumptions or structural assumptions –, there exists a structure to be learnt that makes the problem at hand much simpler in the sense that it requires a reduced amount of samples per task to solve. This is typical of *meta-learning* or *learning-to-learn* approaches in deep-learning, where the process of learning itself is being improved by multiple learning episodes (that consist of the various users' task in our case) [HAMS20]. Whether a cluster structure, dissimilarity between clients, or a shared representation is known before solving each client task, the whole procedure becomes much easier.

The contributions of this thesis in these directions are threefold. We formalize the problem of learning personalized models as generalization and stochastic optimization problems in Chapters 8 and 9, under respectively user-wise structures or model-wise structures. We study the problem of jointly learning n models for n agents, based on the local datasets of these users, while making the kind of assumption described above. More formally, each agent i indexed by $i \in [n]$ has a local data distribution $\mathcal{D}_i$, and aims at minimizing its *generalization error*, defined as for some loss $F_i : \mathbb{R}^d \times \Xi \to \mathbb{R}$:

$$f_i(x) = \mathbb{E}\left[F_i(x, \xi_i)\right], \quad x \in \mathbb{R}^d, \quad \xi_i \sim \mathcal{D}_i .$$

In order to capture the minimization of each local function and thus *personalization*, the objective we want to minimize then writes as:

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x_i), \quad x = (x_1 | \dots | x_n) \in \mathbb{R}^{d \times n} . \tag{1.12}$$

We will consider two settings in the two chapters of this Part: the online setting and the classical supervised learning setting, respectively in Chapters 8 and 9.

In the online setting, data samples come one at a time for each agent. At iteration $k \geq 0$, all agents $i \in [n]$ access a new sample ξ_i^k, independent from the previous ones, and may update its local model using its own information, and information that other agents may have shared. The *no-collaboration* baseline in this setting would then be the online SGD algorithm:

$$\forall i \in [n], \quad \forall k \geq 0, \quad x_i^{k+1} = x_i^k - \eta_k \nabla_x F_i(x_i^k, \xi_i^k),$$

while the full collaboration baseline is Minibatch online SGD, for which $x_i^k = x_j^k$ for all k:

$$\forall i \in [n], \quad \forall k \geq 0, \quad x_i^{k+1} = x_i^k - \frac{\eta_k}{n} \sum_{i=1}^{n} \nabla_x F_i(x_i^k, \xi_i^k) .$$

This latter algorithm benefits from smaller variance, but may be highly biased on the local of of users. The algorithms we will consider in Chapter 8 are a midway between these two opposite paradigms.

In the classical supervised learning setting, each agent has a local dataset of size m, $\{\xi_{ij}\}_{j \in [m]}$ with $\xi_{ij} \sim \mathcal{D}_i$. Based on the m samples per agent we have, agents will minimize their empirical risks using collaboration to increase their sample efficiency. We denote as $\mathcal{L}_i$

the empirical risk of agent i, that writes:

$$\mathcal{L}_i(x) = \frac{1}{m}\sum_{j=1}^{m} F_i(x, \xi_{ij})\,, \quad x \in \mathbb{R}^d\,,$$

and as $\mathcal{L}$ the global empirical risk:

$$\mathcal{L}(x) = \frac{1}{nm}\sum_{(i,j)\in[n]\times[m]} F_i(x_i, \xi_{ij})\,, \quad x = (x_1|\dots|x_n) \in \mathbb{R}^{d\times n}\,. \tag{1.13}$$

By minimizing this empirical risk, we wish to obtain good minimizers for the true test loss (Equation (9.1)), under adequate assumptions. The no-collaboration baseline estimator here writes as

$$\forall i \in [n]\,, \quad \hat{x}_i \in \arg\min_{x\in\mathbb{R}^d} \mathcal{L}_i(x)\,,$$

while the full collaboration baseline is:

$$\forall i \in [n]\,, \quad \hat{x}_i \in \arg\min_{x\in\mathbb{R}^d} \mathcal{L}(x)\,,$$

which benefits from all nm samples, but might poorly generalize on local users data distributions. The goal is thus to regularize this estimator to enforce some form of collaboration. In both cases, counting k the number of steps to reach precision ε for local objectives or counting m the number of local data samples required to obtain generalization error ε both amount to computing the *sample complexity* required for collaboratively minimizing generalization errors. For our problem, the structural assumptions described above then read as follows. Note that Assumption 1.4.3 is in fact a subcase of Assumptions 1.4.2 and 1.4.1.

Assumption 1.4.1 (Similarity assumption)**.** *There exist non-negative weights $(b_{ij})_{i,j\in[n]}$ and minimizers $(x_i^\star)_{i\in[n]}$ of functions $(f_i)_{i\in[n]}$ such that for all $i, j \in [n]$:*

$$f_i(x_j^\star) - f_i(x_i^\star) \le b_{ij}\,.$$

Assumption 1.4.2 (Low rank representation)**.** *There exist $\{w_i^\star\}_{i\in[n]}$ such that for all i $w_i^\star$ minimizes f_i, and the rank of the matrix $W^\star = (w_1^\star|\dots|w_n^\star) \in \mathbb{R}^{d\times n}$ is at most r. Equivalently, there exist an orthonormal matrix $U_\star \in \mathbb{R}^{d\times r}$ and $V^\star = (v_1^\star, \dots, v_n^\star) \in \mathbb{R}^{r\times n}$ such that $w_i^\star = U_\star v_i^\star$ for all $i \in [n]$.*

Assumption 1.4.3 (Clustered clients)**.** *There exist $\{w_i^\star\}_{i\in[n]}$, $\{c_s^\star\}_{s\in[r]}$ and $\tau^\star : [n] \to [r]$ such that for all i, $w_i^\star$ minimizes f_i and $w_i^\star = c^\star_{\tau^\star(i)}$.*

Contributions of Chapters 8 and 9. The purpose of these two chapters is to study natural minimizers under these assumptions, for convex objectives f_i. In Chapter 8 we study the online setting under Assumption 1.4.1: we provide information theoretical lower bounds for the problem and online algorithms that have performances matching these lower bounds. Under assumptions on users similarity (Assumption 1.4.1), we first provide information-theoretical lower bounds, that depend on some distance dist between agents distributions $\mathcal{D}_i$ (Figure 1.6 – under our assumptions, $\mathrm{dist}(\mathcal{D}_i, \mathcal{D}_j) = b_{ij}$ here). Formally, for some given precision ε, let $G_\varepsilon = (\mathcal{V}, \mathcal{E}_\varepsilon)$ be the graph on $\mathcal{V} = [n]$ with edges $\{i, j\} \in \mathcal{E}_\varepsilon$ iff $\mathrm{dist}(\mathcal{D}_i, \mathcal{D}_j) \le \varepsilon$. Then, denoting as $\mathcal{N}_\varepsilon(i)$ the number of neighbors of agent i in this graph, we show that the statistical collaboration speedup cannot be larger than $\frac{1}{\mathcal{N}_\varepsilon(i)}$ for agent i, therefore confirming the bias-variance tradeoff intuited above. In Chapter 9 we study the supervised setting under Assumptions 1.4.2 or 1.4.3.

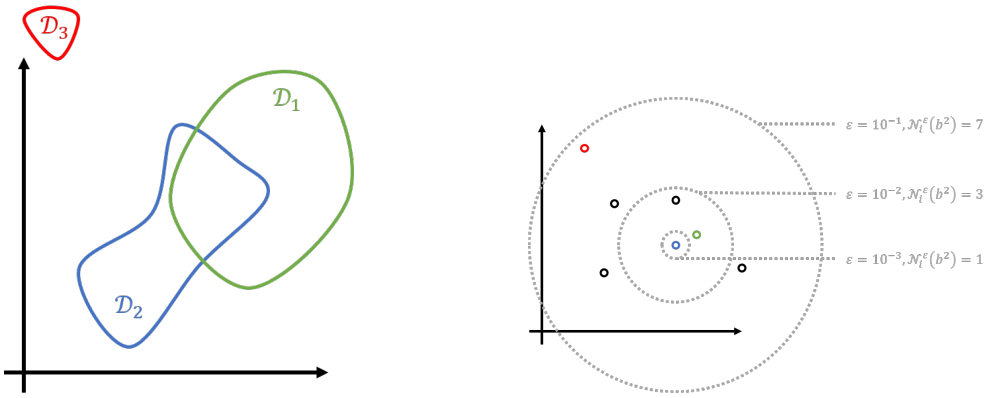


Figure 1.6 – *Quantities $\mathcal{N}_\varepsilon$ for different distributions and illustration of the similarity graph.*

1.5. Private learning

Finally, we introduce the notion of privacy in ML and in particular in distributed learning. When dealing with users data — that may range from personal data held in cellphones that can contain passwords, contacts, private messages or bank references, to hospital clinical data —, for the users to be compliant for their data to be used (or at least for users to be not too much against their use), it is necessary to ensure that their private informations are not released to the public.

This is where privacy comes in: privacy can be quantified, strategies that ensure that data remains private when training models may be quantitatively evaluated in terms of their privacy preserving properties. But first, from where can data be leaked ? From the ML algorithms presented previously in this introduction (Algorithms 1.1 to 1.3), their common ground is that what is being shared to an eventual attacker (the server, someone that may intercept communications, or someone who sees the model in the end) are *gradients*, of the form $\nabla F(x, (a, b))$, computed at some model $x \in \mathbb{R}^d$, for some data input and label a, b. Data samples are thus not shared, and these algorithms are thus naturally private — at least that is what we would ideally hope for.

There is however a twist: data points can be inferred from gradients, a phenomenon referred to as *deep leakage from gradients* [ZH20]. This can in fact be done quite easily, through an inverse problem formulation. Upon reception of some stochastic gradient $G = \nabla F(x, (a^\star, b^\star))$ computed at some model x (known by the attacker, if we assume weights are shared) for some datapoints $a^\star, b^\star$ to unveil, the attacker solves the following inverse problem:

$$(\hat{a}, \hat{b}) \in \arg\min_{(a,b)} \|\nabla F(x, (a, b)) - G\|^2 , \tag{1.14}$$

using for instance gradient descent (that requires second order information). Such attacks prove to be quite effective in practice, and show that even if we think that we do not share our information, it can still be retrieved in unexpected ways.

We thus have to ensure that data remains private, *whatever an attacker might be able to do*, a solution brought by *Differential Privacy*. In Equation (1.14), the attacker sees a function $\mathcal{A}(\mathcal{D}_n)$: what an algorithm (here SGD or the intermediate steps of SGD) outputs when fed with data samples $\mathcal{D}_n = \{(a_i, b_i), i \in [n]\}$. This algorithm $\mathcal{A}$ is usually random: $\mathcal{A}(\mathcal{D}_n)$ is thus a random variable. We know for sure that, if for any (a'_1, b'_1), when data sample (a_1, b_1) is replaced (a'_1, b'_1) (leading to data set $\mathcal{D}'_n$ instead of $\mathcal{D}_n$: these datasets differ only on one data sample) we have $\mathcal{A}(\mathcal{D}_n) \approx \mathcal{A}(\mathcal{D}'_n)$, then $\mathcal{A}$ does not see the difference between whatever we put in the first sample: this sample is thus kept private and any strategy such as Equation (1.14) will provably fail.

This intuition is formalized by Differential Privacy.

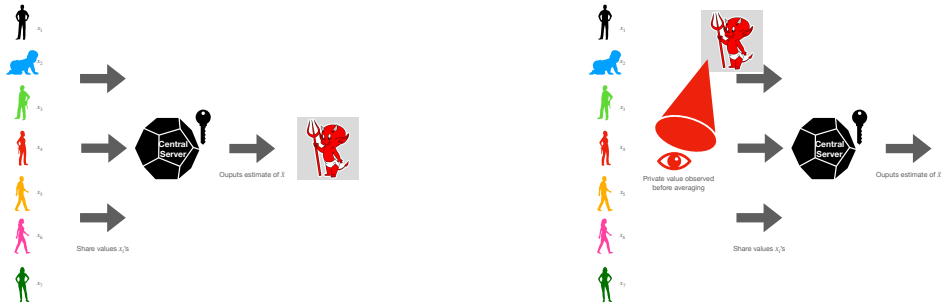


Figure 1.7 – *Central DP (left) and Local DP (right)*

Definition 1.5.1 (Differential Privacy)**.** *Let $\varepsilon > 0$. An algorithm $\mathcal{A}$ satisfies $\varepsilon-$Differential Privacy ($\varepsilon-$DP) if for all pairs of neighboring datasets $\mathcal{D} \sim \mathcal{D}'$* (i.e., *datasets that differ on only on data sample), for any event S, we have:*

$$\mathbb{P}\left(\mathcal{A}(\mathcal{D}') \in S\right) \leq e^{\varepsilon}\mathbb{P}\left(\mathcal{A}(\mathcal{D}) \in S\right),$$

or equivalently, if $\frac{\mathrm{d}\mathbb{P}'}{\mathrm{d}\mathbb{P}}$ (where p' and p are respectively the probability laws of $\mathcal{A}(\mathcal{D})$ and $\mathcal{A}(\mathcal{D}')$) satisfies:

$$\log\left(\frac{\mathrm{d}\mathbb{P}'}{\mathrm{d}\mathbb{P}}\right) \leq \varepsilon \qquad \textit{almost surely.}$$

Several extensions were developed to relax the definition of DP ($(\varepsilon, \delta)-$DP, Rényi DP), but they all share the same objective. The relaxation we use in our manuscript is the following: it bounds a milder divergence than the log density.

Definition 1.5.2 (Rényi Differential Privacy)**.** *An algorithm $\mathcal{A}$ satisfies (α, ε)-Rényi Differential Privacy (RDP) for $\alpha > 1$ and $\varepsilon > 0$ if for all pairs of neighboring datasets $\mathcal{D} \sim \mathcal{D}'$:*

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

where for two random variables X and Y, $D_\alpha\big(X \,||\, Y\big)$ is the Rényi divergence *between X and Y:*

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

with μ_X and μ_Y the respective densities of X and Y.

Private mean estimation as a toy problem. When introducing decentralized algorithms and in particular gossiping, we first presented the averaging problem. This is again what we are going to do here, by introducing the private mean estimation problem. Suppose our n agents each hold private values $x_i \in \mathbb{R}^d$, and that we want to compute the average $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i$ in a *private* way, without revealing the x_i's. Then, in order to be private, instead of sending the values x_i's to the server, users generate some random noise and send $\tilde{x}_i \sim \mathcal{N}(x_i, \sigma^2 I_d)$ to the server. The server then outputs $\hat{x} = \frac{1}{n}\sum_{i=1}^n \tilde{x}_i$. Adding Gaussian noise is what we call the *Gaussian mechanism.*

Proposition 1.5.1 (Gaussian mechanism, [Mir17])**.** *Let $\mathbf{G}_\sigma$ be the Gaussian mechanism, defined as $\mathbf{G}_\sigma\mathcal{A}(\mathcal{D}) = \mathcal{A}(\mathcal{D}) + \mathcal{N}(0, \sigma^2 I_d)$, for any algorithm $\mathcal{A}$ with output values in $\mathbb{R}^d$ and any input sample set $\mathcal{D}$. Define the ℓ^2-sensitivity of $\mathcal{A}$ as:*

$$\Delta_2(\mathcal{A}) = \sup_{\mathcal{D}\sim\mathcal{D}'} \left\|\mathcal{A}(\mathcal{D}) - \mathcal{A}(\mathcal{D}')\right\|_2,$$

where the sup is taken over all adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$. Then, $\mathbf{G}_\sigma\mathcal{A}$ is $(\alpha, \frac{\alpha\Delta_2(\mathcal{A})^2}{2\sigma^2})-$Rényi differentially private.

Now, as illustrated in Figure 1.7, the level of privacy guaranteed by the added noise depends on where the *malicious* node/the attacker is (the little devil in Figure 1.7): is the server the attacker ? Or does the attacker only observes the output $\hat{x}$? In the first case, the attacker sees all the values $\tilde{x}_i$ and we refer to this as *Local DP*, while in the second case only their average is observed, a case referred to as *Central DP*. The latter is of course more private, and consists in assuming the existence of a *central curator*.

In local DP, we have $\mathcal{A}(x_1, \dots, x_n) = (\tilde{x}_1, \dots, \tilde{x}_n) = \mathbf{G}_\sigma(x_1, \dots, \tilde{x}_n)$, while in central DP we instead have $\mathcal{A}(x_1, \dots, x_n) = \hat{x} = (\tilde{x}_1 + \dots + \tilde{x}_n)/n = \mathbf{G}_{\frac{\sigma}{\sqrt{n}}}(\bar{x})$. The sensitivities of these two algorithms differ: if all x_i's are restricted to live in a ball of diameter Δ, then for Local DP the sensitivity is Δ, while it is Δ/n in Central DP. The ratios sentivity squared divided by Gaussian noise variance squared are thus respectively equal to $\frac{\Delta^2}{\sigma^2}$ and $\frac{\Delta^2}{n\sigma^2}$. Overall, optimizing over σ^2 under RDP constraint, to achieve (α, ε)-RDP, one cannot have better utility than:

$$\mathbb{E}\left[\|\hat{x} - \bar{x}\|^2\right] \leq \frac{\alpha\Delta^2}{2n\varepsilon} \quad \text{for local DP}\,,$$
$$\mathbb{E}\left[\|\hat{x} - \bar{x}\|^2\right] \leq \frac{\alpha\Delta^2}{2n^2\varepsilon} \quad \text{for central DP}\,,$$

where $\hat{x}$ is the output of the algorithm. This $1/n$ gap motivates the study of relaxations of local DP. Of course, these notions of local and central DP are not specific to private mean estimation: they directly extend to learning in general, with algorithms such as DP-SGD [ACG+16, Differentially Private SGD] or other variants: noise is always added in order to satisfy some DP properties.

What about decentralized algorithms ? There is a huge gap between the performances in the central DP and local DP settings, given a fixed $(\alpha, \varepsilon)-$RDP budget. As we saw in previous sections, there is no central curator in decentralized learning. Agents communicate with each other, and in worst cases, when gossiping to compute private means, the neighbor of a given agent is malicious. Hence, we cannot hope for better privacy guarantees than Local DP, if some agents are too curious: this is a shame, since decentralized algorithms are known for being more privacy preserving than their centralized counterparts, due to the fact that communications are not centralized to a server that might be malicious.

Even though users that are neighbors to malicious nodes cannot apriori do better than Local DP, since the malicious agent then receives directly all that this node shares, if a node is further away in the graph, it should be better protected against attacks.

In the Harry Potter series, J.K Rowling introduces *Muffliato*, invented by Severus Snape [Sna46] in the margin of its personal version of Libatius Borage's book on potion making [Bor76]: it designates a "spell that filled the ears of anyone nearby with an unidentifiable buzzing", thereby concealing messages from unintended listeners through noise injection. Consider now the gossip algorithm as formulated in Equation (1.8), but initialized at noisy values $\tilde{x}_i \sim \mathcal{N}(x_i, \sigma^2 I_d)$ instead of x_i, in order to compute $\hat{x}$ private estimate of $\bar{x}$ in a decentralized way. Each node add some noise at initialization — some local *buzzing* —, and two nodes that are far away from each other then hear all the buzzing from all the nodes that are on the paths between these two nodes: gossiping with noisy initial values behaves exactly like the spell *Muffliato*!

Contributions of Chapter 6. We introduce a relaxation of Local and Central DP for decentralized learning, coined as pairwise network differential privacy. This new notion of privacy accounts for the positions of nodes in the communication graph, enabling us to argue that, for any two nodes u, v, roughly, *"node u is private with respect to node v with a privacy loss*

quantified by $\varepsilon_{u\to v}$", where the quantity $\varepsilon_{u\to v}$ decreases as the distance between nodes u and v in the graph increases.

Our notion of privacy thus captures the impact of decentralization in a very natural way, and the *Muffliato* block — noise injection with gossip communications — can be added to decentralized learning algorithms like Decentralized SGD, leading to private decentralized learning, that then provably benefits from improved privacy guaratantees.

1.6. Additional assumptions and notations used throughout the manuscript

1.6.1. Notations

In the manuscript, we make use of many notations for matrices and sets of matrices. The most classical one is $A \in \mathbb{R}^{p\times q}$: A is a matrix with p rows and q columns. We make also use of more abstract notations. For $\mathcal{V}$ and $\mathcal{E}$ finite sets (that usually correspond to graph node and edge sets), we may for instance write $A \in \mathbb{R}^{\mathcal{V}\times d}$ for $A \in \mathbb{R}^{n\times d}$ where $n = |\mathcal{V}|$, except that if $\mathcal{V} \neq [n]$,the lines of A are indexed by the set $\mathcal{V}$. Notations $\mathbb{R}^{\mathcal{E}\times d}$ or $\mathbb{R}^{\mathcal{E}\times\mathcal{V}}$ can also be used in the same way.

For two symmetric matrices $A, B \in \mathbb{R}^{p\times p}$, the notation $A \succeq B$ (resp. $A \preceq B$) means that $A-B$ is positive semi-definite *i.e.* all its eigenvalues are non negative (resp. $B-A$ is positive semi-definite).

For some vector $u \in \mathbb{R}^p$, $\|u\| = \sqrt{\sum_{k=1}^p u_k^2}$ always refers to the ℓ^2 norm, $\|u\|_p = \left(\sum_{k=1}^p |u_k|^p\right)^{\frac{1}{p}}$, $\|u\|_\infty = \max_{k=1,\dots,p} |u_k|$ and $\|u\|_0 = |\{k \in [p] : u_k \neq 0\}|$. For two vectors $u, v \in \mathbb{R}^p$, $\langle u, v\rangle = \sum_{k=1}^p u_k v_k$ is their scalar product. For matrix $A \in \mathbb{R}^{p\times q}$, its *singular value decomposition* (SVD) is defined as $A = P_1 D P_2^\top$ where $P_1 \in \mathbb{R}^{p\times r}$ and $P_2 \in \mathbb{R}^{q\times r}$ are matrices with orthogonal columns, where $r = \min(p, q)$, and $D = \mathrm{diag}(\lambda_1, \dots, \lambda_{\mathrm{r}}) \in \mathbb{R}^{\mathrm{r}\times\mathrm{r}}$ is a diagonal matrix, with diagonal coefficients $\lambda_1 \geq \dots \geq \lambda_r \geq 0$, that are the *singular values* of A. The Frobenius norm of A writes as $\|A\|_F = \sqrt{\sum_{k=1}^r \lambda_k^2} = \sqrt{\sum_{(k,\ell)\in[p]\times[q]} A_{k\ell}^2}$, its nuclear norm $\|A\|_* = \sum_{k=1}^r \lambda_k$, and its operator norm $\|A\|_{\mathrm{op}} = \lambda_1$. For two matrices $A, B \in \mathbb{R}^{p\times q}$, $\langle A, B\rangle = \sum_{(k,\ell)\in[p]\times[q]} A_{k\ell} B_{k\ell}$ is their scalar product.

1.6.2. Various regularity assumptions

Let $f : \mathbb{R}^d \to \mathbb{R}$. We here recall definitions related to the regularity of f that will be used throughout the manuscript, and that are presented more thoroughly in [Bub15] for instance.

Definition 1.6.1 (Convexity)**.** *f is convex if for all $x, y \in \mathbb{R}^d$ and $\lambda \in [0, 1]$, we have $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$. Equivalently, if f is differentiable, it is convex if and only if its curve is above all its cords: $\forall x, y \in \mathbb{R}^d$, $f(x) \geq f(y) + \langle \nabla f(y), x - y\rangle$. If f is twice differentiable, f is convex if and only if $\nabla^2 f(x) \succeq 0$ for all $x \in \mathbb{R}^d$.*

Definition 1.6.2 (Lipschitz)**.** *A function $g : \mathbb{R}^p \to \mathbb{R}^q$ is $M-$Lipschitz if for all $x, y \in \mathbb{R}^p$, we have $\|g(x) - g(y)\| \leq \|x - y\|$. If g is differentiable, this is equivalent to $\|\nabla g(x)\|_{\mathrm{op}} \leq M$ for all $x \in \mathbb{R}^d$.*

Definition 1.6.3 (Strong convexity and smoothness)**.** *f is $\mu-$strongly convex and $L-$smooth for some $0 \leq \mu \leq L$ if:*

$$\forall x, y \in \mathbb{R}^d\,, \quad \frac{\mu}{2}\|x - y\|^2 \leq D_f(x, y) \leq \frac{L}{2}\|x - y\|^2\,, \tag{1.15}$$

where D_f is the Bregman divergence of f, defined as:

$$D_f(x, y) = f(x) - f(y) - \langle \nabla f(y), x - y\rangle\,.$$

If f is twice differentiable, Equation (1.15) is equivalent to:

$$\forall x, y \in \mathbb{R}^d\,, \quad \mu I_d \preceq \nabla^2 f(x) \preceq L I_d\,.$$

Definition 1.6.4 (Fenchel Conjugate)**.** *The* Fenchel conjugate *of f is denoted by f^* and defined on $\mathbb{R}^d$ by*

$$f^*(y) = \sup_{x \in \mathbb{R}^d} \{\langle x, y\rangle - f(x)\} \in \mathbb{R} \cup \{+\infty\}\,.$$

An important property of Fenchel conjugates is their link with Bregman divergence. If f is differentiable, then for all $x, y \in \mathbb{R}^d$, we have:

$$D_f(x, y) = D_{f^*}(\nabla f(y), \nabla f(x))\,.$$

As a consequence, if f is $\mu-$strongly convex and $L-$smooth, then f^* is $1/L-$strongly convex and $1/\mu-$smooth. Furthermore, this enables us to prove that under strong convexity and smoothness assumption, we have:

$$\forall x \in \mathbb{R}^d\,, \quad 2\mu(f(x) - f(x^\star)) \leq \|\nabla f(x)\|^2 \leq 2L(f(x) - f(x^\star))\,,$$

where $x^\star$ is a minimizer of f. The LHS of the above inequality is a $\mu-$Polyak-Łojasiewicz inequality.

Definition 1.6.5 ($\mu-$Polyak-Łojasiewicz)**.** *f satisfies the $\mu-$Polyak-Łojasiewicz inequality if it is minimized at some $x^\star \in \mathbb{R}^d$ and:*

$$\forall x \in \mathbb{R}^d\,, \quad \|\nabla f(x)\|^2 \geq 2\mu(f(x) - f(x^\star))\,.$$

When objective f is (strongly) convex, we will be able to obtain an upper bound on the expected suboptimality gradient-based algorithm's output $\tilde{x}$ of the form $\mathbb{E}f(\tilde{x}) - f^* \leq \varepsilon$ for some explicit ε. On the other hand, when the objective is not convex, it is generally intractable to approximate global minima [NY83] so, as is common in the literature, we will generally fall back to showing that the algorithm will find an approximate first-order stationary point of the objective: $\mathbb{E}\|\nabla f(\tilde{x})\|^2 \leq \varepsilon^2$.

Part I

Asynchronous communications and computations

Asynchronous optimization and decentralized learning

Asynchronous optimization. Asynchronous optimization has a long history. In the 1970s, [Bau78] considered shared-memory asynchronous fixed-point iterations, and an early convergence result for Asynchronous SGD was established by [TBA86]. Recent analysis typically relies on bounded delays [AD11, RRWN11, LHLL15, SK20]. [SYLS16] slightly relax this to random delays with bounded expectation. [ZMB+18] allowed delays to grow over time, but only show asymptotic convergence. Some algorithms try to adapt to the delays, but even these are not proven to perform well under arbitrary delays [ZMW+17b, MIMA18]. For more examples of stochastic asynchronous algorithms, we refer readers to the surveys by [BNH19, AAF+20]. The goal of Chapter 3 is specifically to have guarantees that are delay-independent for Asynchronous SGD, and to obtain a wall-clock time asynchronous speedup.

In the online learning setting, [MS14, JGS16] studied an adaptive asynchronous SGD algorithm. [AHSL21] proved better guarantees on bounded domains by introducing a projection, with rates depending on the average rather than maximum delay. However, their proof relies heavily on the assumption of a bounded domain, while ours applies for optimization over all of $\mathbb{R}^d$. Relatedly, [CDD+21] prove guarantees for Asynchronous SGD that depend on the average delay, but their results only hold with probability $\frac{1}{2}$.

[MPP+17] proposed and utilized the analysis tool of "virtual iterates" for Asynchronous SGD under bounded delays. [SK20] extended these results, albeit restricting delays to be constant, and [LPLJ18] considered lock-free updates. We use a similar proof approach in Chapters 3 and 5, but with different virtual sequences and eventually different, delay-adaptive stepsizes.

Decentralized optimization. Gossip algorithms [BGPS06, DKM+10] were initially introduced to compute the global average of local vectors with local pairwise communications only (no central coordinator), and were generalized to decentralized optimization. Two types of gossip algorithms appear in the literature: synchronous ones, where all nodes communicate with each other simultaneously [DKM+10, SBB+17, KSJ19, SLWY15], and randomized ones [BGPS06, NO09]. A third category considers directed (non-symmetric) communication graphs [XMX+18, AR21] which are much easier to implement asynchronously. In the synchronous framework, the communication speed is limited by the slowest node (*straggler* problem), whereas the classical randomized gossip framework of [BGPS06] assumes communications to happen instantaneously, and thus does not address the question of how to deal with delays. Decentralized SGD [KSJ22, LZZ+17, e.g.] consists in iterations where at every time step, all nodes perform local SGD steps, and communicate their local model with their neighbors in a graph (that may vary with time, but that needs to mix in an ergodic way).

Decentralized optimization and asynchrony. Combining both decentralization and asynchrony is a challenging problem, and it is only recently that this question has risen a surge of interest [AR21, BRW+23, LHZQ20, LYW+22, NSD+21, ZY21]. These works are however restricted to a given communication protocol and static topologies under an *i.i.d.* sampling of the nodes or edges that become active [AR21, LHLL15, BRW+23, NSD+21], no communication delays [LHLL15, BRW+23, NSD+21], or their analyses rely on an upper-bound on the maximal computation and communication delays [AR21, LZZL18, BRW+23, LHZQ20, LYW+22, NSD+21, ZY21, WLMJ23]. This latter point is exactly the goal of Chapters 4 and 5: how can we relax the worst case delay dependency and capture quantitatively delay heterogeneity in the graph and relate it to a *wall-clock asynchronous speedup*? Similarly to the asynchronous speedup of Asynchronous SGD described just above, we will quantify the asynchronous speedup as a graph dependent quantity that takes into account pairwise communication delays and their heterogeneity.

Chapter 2

The continuized framework: continuized Nesterov acceleration and acceleration of randomized gossip

In this chapter, we introduce the "continuized" framework, that consists of an alternative to both discrete algorithmic frameworks under which gradient-based algorithms are usually studied and continuous-time frameworks used in the analysis of gradient flows for instance. This framework will then be used in subsequent chapters for the design of aynchronous algorithms. The idea behind this framework is to combine the best of both discrete and continuous worlds: the analysis-friendiness and the elegance of continuous time analyses together with the direct implementability of discrete algorithms. As a first application of this framework, we provide the continuized Nesterov acceleration is a close variant of Nesterov acceleration whose variables are indexed by a continuous time parameter. The two variables continuously mix following a linear ordinary differential equation and take gradient steps at random times. As a continuous process, one can use differential calculus to analyze convergence and obtain analytical expressions for the parameters. A discretization of the continuized process can be computed exactly with convergence rates similar to those of Nesterov original acceleration. We show that the discretization has the same structure as Nesterov acceleration, but with random parameters. We provide continuized Nesterov acceleration under deterministic as well as stochastic gradients, with either additive or multiplicative noise.

Finally, using our continuized framework and expressing the gossip averaging problem as the stochastic minimization of a certain energy function, we provide the first rigorous acceleration of randomized gossip algorithms. In randomized gossip, nodes in a graph $G = (\mathcal{V}, \mathcal{E})$ perform pairwise updates of the form

$$x_v, x_w \longleftarrow \frac{x_v + x_w}{2}$$

at random Poisson clock tickings. The acceleration of randomized gossip we propose consists of similar updates at random Poisson clock tickings, together with an underlying continuous dynamics that allows faster mixing.

2.1. Introduction

In the last decades, the emergence of numerous applications in statistics, machine learning and signal processing has led to a renewed interest in first-order optimization methods [BCN18]. They enjoy a low iteration cost necessary to the analysis of large datasets. The performance of first-order methods was largely improved thanks to acceleration techniques (see the review by [dST21] and the many references therein), starting with the seminal work of [Nes83].

Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex and differentiable function, minimized at $x_* \in \mathbb{R}^d$. We assume

throughout this chapter that f is L-smooth (Definition 1.6.3) and convex. In addition, we will sometimes also assume that f is μ-strongly convex for some $\mu > 0$. For the problem of minimizing f, gradient descent is well-known to achieve a rate $f(x_k) - f(x_*) = O(k^{-1})$ in the smooth case, and a rate $f(x_k) - f(x_*) = O((1 - \mu/L)^k)$ in the smooth and strongly convex case. In both cases, Nesterov introduced an alternative method with essentially the same iteration cost, while achieving faster rates: it converges with rate $O(k^{-2})$ in the smooth convex case and with rate $O((1 - \sqrt{\mu/L})^k)$ in the smooth and strongly convex case [Nes03]. These rates are then optimal among all black-box first-order methods that access gradients and linearly combine them [Nes03, NY83].

Nesterov acceleration relies on several sequences of iterates—two or three, depending on the formulation—and on a clever blend of gradient steps and mixing steps between the sequences. Different interpretations and motivations underlying the precise structure of accelerated schemes were approached in many works, including [BLS15, FB15, ASSS16, KF16, AO17]. A large number of these works studied continuous time equivalents of Nesterov acceleration, obtained by taking the limit when stepsizes vanish, or from a variational framework. The continuous time index t of the limit allowed to use differential calculus to study the convergence of these equivalents. Examples of studies relying on continuous time interpretation include [SBC14, KBB15, WRJ16, WWJ16, BJW18, DO19, SDJS18, SDSJ19, ACPR18, ACR19, ZMSJ18, MJ19].

Continuized Nesterov acceleration. In this chapter, we propose another continuous time equivalent to Nesterov acceleration, which we refer to as the *continuized* Nesterov acceleration, which avoids vanishing stepsizes. It is built by considering two sequences $x_t, z_t \in \mathbb{R}^d$, $t \in \mathbb{R}_{\geq 0}$, that continuously mix following a linear ordinary differential equation (ODE), and that take gradient steps at random times $T_1, T_2, T_3, \dots$. Thus, in this modeling, mixing and gradient steps alternate randomly.

Thanks to the continuous index t and some stochastic calculus, one can differentiate averaged quantities (expectations) with respect to t. In particular, this leads to simple analytical expressions for the optimal parameters as functions of t, while the optimal parameters of Nesterov accelerations are defined by recurrence relations that are complicated to solve.

The discretization $\tilde{x}_k = x_{T_k}, \tilde{z}_k = z_{T_k}, k \in \mathbb{N}$, of the continuized process can be computed directly and exactly: the result is a recursion of the same form as Nesterov iteration, but with randomized parameters, and performs similarly to Nesterov original deterministic version both in theory and in simulations.

The continuized framework can be adapted to various settings and extensions of Nesterov acceleration. In what follows, we study how the continuized acceleration behaves in the presence of *additive* and *multiplicative* noise in the gradients. In the multiplicative noise setting, our acceleration satisfies a convergence rate similar to that of [JKK+18] and depends on the *statistical condition number* of the problem at hand. The two acceleration schemes are not directly comparable as we work in a continuized setting and only deal with pure multiplicative noise. Our analysis is nevertheless much simpler, as it closely mimics that of Nesterov acceleration.

Application to accelerated gossip algorithms. The continuized modeling is natural in asynchronous parallel computing where gradient steps arrive at random times. More importantly, there are situations where the continuized version of Nesterov acceleration can be practically implemented while the original acceleration can not. In distributed settings, for instance, the total number k of gradient steps taken in the network is typically not known to each particular node; an advantage of the continuized acceleration is that it requires to know only the time t and not k.

Gossip algorithms typically feature such asynchronous and distributed behaviors [BGPS06]. In gossip problems, nodes of a network aim at computing the global average of all their values

by communicating only locally (with their neighbors), and without centralized coordination. In this set-up, pairs of adjacent nodes communicate at random times, asynchronously, and in parallel, so that the total number of past communications in the network at a given time is unknown to all nodes. In this chapter, we formulate the gossip problem as a stochastic optimization problem. Thanks to the continuized formalism, we naturally obtain accelerated gossip algorithms that can be implemented in an asynchronous and distributed fashion.

Synchronous gossip algorithms rely on all nodes to communicate simultaneously [DKM+10]. Accelerating synchronous gossip algorithms have been studied in previous works, including *SSDA* [SBB+17], Chebyshev acceleration [MMS11], Jacobi-Polynomial acceleration [BBG20b]. To that day, acceleration in the asynchronous setting has also been studied in a few works (see for instance geographic gossip [DSW08], shift registers [LACM13], *ESDAC* [HBM18], and randomized Kaczmarz methods [LRR19]). However, no algorithm in an asynchronous framework has been rigorously proven to achieve an accelerated rate for general graphs [DSW08]. Other acceleration schemes [HBM18, LRR19] relied on additional synchronizations between nodes, such as the knowledge of a global iteration counter. This departs from purely asynchronous operations, hence leading to practical limitation. Our accelerated randomized gossip algorithm recovers the same accelerated rates of [HBM18], and only requires the knowledge of a common continuous-time $t \in \mathbb{R}_{\geq 0}$.

In this context, the continuized acceleration should be seen as a close approximation to Nesterov acceleration, that features both an insightful and convenient expression as a continuous time process and a direct implementation as a discrete iteration. We thus hope to contribute to the understanding of Nesterov acceleration. In practice, the continuized framework is relevant for handling asynchrony in decentralized optimization, where agents of a network can not share a global iteration counter, preventing accelerated decentralized and asynchronous methods.

Notations in the continuized framework. The index k always denotes a non-negative integer, while indices t, s always denote non-negative reals.

2.2. The continuized framework

Let $(X_k)_{k\in\mathbb{N}}$ be a Markov chain on a discrete space $\mathcal{X}$, of probability transition matrix $P \in [0,1]^{\mathcal{X}\times\mathcal{X}}$: for all $k \geq 0$ and all $x_0, \dots, x_k, x_{k+1} \in \mathcal{X}$, we have $\mathbb{P}\left(X_{k+1} = x_{k+1} \,|\, x_0, \dots, x_k\right) = P_{x_k, x_{k+1}}$. [AF02] introduced the *continuized* version of the Markov chain, the continuous-time process $(\tilde{X}(t))_{t\in\mathbb{R}_{\geq 0}}$ such that there exist random times $0 = T_0 < T_1 < \dots$ such that for all $t \geq 0$ we have $\tilde{X}(t) = X_k$ for all $t \in [T_k, T_{k+1})$. The random times $0 = T_0 < T_1 < \dots$ are usually a *Poisson point process*: $T_1, T_2 - T_1, T_3 - T_2, \dots$ are independent identically distributed (i.i.d.), of law exponential with rate 1.

Now, let us consider an *algorithm* $\mathcal{A}$ (*e.g.*, gradient descent iterates) that output a discrete sequence generated through iterations $A_{k+1} = F_k(A_0, \dots, A_k)$, where F_k might be random. The continuized version of $\mathcal{A}$ is the continuous-time sequence $\tilde{A}(t)$ satisfying $\tilde{A}(t) = A_k$ for all $t \in [T_k, T_{k+1})$ where $0 = T_0 < T_1 < \dots$ are random times. We here see that this continuous-time sequence can be cast as $\tilde{A}(T_{k+1}) = F_k(\tilde{A}_{T_0}, \dots, \tilde{A}_{T_k})$.

Up to now, the continuized version of $\mathcal{A}$ does not bring new information, since continuous-time is not used so far. However, as we will see in the following sections, iterations $\tilde{A}(T_{k+1}) = F_k(\tilde{A}_{T_0}, \dots, \tilde{A}_{T_k})$ can be generalized to obtain continuized iterates of the form:

$$\forall t \notin \{T_0, T_1, \dots\}, \quad \mathrm{d}\tilde{X}(t) = \phi\left(\left\{\tilde{X}(s), s < t\right\}\right)\mathrm{d}t,$$
$$\forall k \in \mathbb{N}, \quad \tilde{X}(T_{k+1}) = \tilde{F}_k\left(\left\{\tilde{X}(s), s < T_{k+1}\right\}\right).$$

Note that this new formulation is strictly equivalent to the discrete one, as some function

F_k and discrete iterates can be deduced from this continuous-time process. However, as illustrated by the continuized version of Nesterov acceleration presented in the subsequent sections, writing continuous processes lead to much simpler analyses. Finally, when discrete processes are harder to implement (usually due to synchronization issues), directly working with the underlying real-world continuous time yields simpler algorithms: this is the approach we take for accelerating randomized gossip in this chapter, and to study delayed randomized gossip in Chapter 4.

2.3. Reminders on Nesterov acceleration

For the sake of comparison, let us first recall the classical Nesterov acceleration. To improve the convergence rate of gradient descent, Nesterov introduced iterations of three sequences, parametrized by $\tau_k, \tau'_k, \gamma_k, \gamma'_k, k \geq 0$, of the form

$$y_k = x_k + \tau_k(z_k - x_k)\,, \tag{2.1}$$
$$x_{k+1} = y_k - \gamma_k \nabla f(y_k)\,, \tag{2.2}$$
$$z_{k+1} = z_k + \tau'_k(y_k - z_k) - \gamma'_k \nabla f(y_k)\,. \tag{2.3}$$

Depending on whether the function f is known to be convex, or strongly convex with a known strong convexity parameter, Nesterov provided a set of parameter choices for achieving acceleration.

Theorem 2.1 (Convergence of accelerated gradient descent)**.** *Nesterov accelerated scheme satisfies:*

1. *Choose the parameters* $\tau_k = 1 - \frac{A_k}{A_{k+1}}, \tau'_k = 0, \gamma_k = \frac{1}{L}, \gamma'_k = \frac{A_{k+1} - A_k}{L}, k \geq 0$*, where the sequence* $A_k, k \geq 0$*, is defined by the recurrence relation*

$$A_0 = 0\,, \qquad\qquad A_{k+1} = A_k + \frac{1}{2}(1 + \sqrt{4A_k + 1})\,.$$

Then

$$f(x_k) - f(x_*) \leq \frac{2L\|x_0 - x_*\|^2}{k^2}\,.$$

2. *Assume further that* f *is* μ*-strongly convex,* $\mu > 0$*. Choose the constant parameters* $\tau_k \equiv \frac{\sqrt{\mu/L}}{1+\sqrt{\mu/L}}$, $\tau'_k \equiv \sqrt{\frac{\mu}{L}}$, $\gamma_k \equiv \frac{1}{L}$, $\gamma'_k \equiv \frac{1}{\sqrt{\mu L}}$, $k \geq 0$*. Then*

$$f(x_k) - f(x_*) \leq \left(f(x_0) - f(x_*) + \frac{\mu}{2}\|z_0 - x_*\|^2\right)\left(1 - \sqrt{\frac{\mu}{L}}\right)^k\,.$$

This result can be found as is in [dST21, Sections 4.4.1 and 4.5.3]. In the convex case, Nesterov acceleration achieves the rate $O(1/k^2)$, whereas gradient descent achieves a rate $O(1/k)$ (see [Nes03, Corollary 2.1.2] for instance). In the strongly convex case, Nesterov acceleration achieves the rate $O((1-\sqrt{\mu/L})^k)$, where gradient descent achieves a rate $O((1-\mu/L)^k)$ (see [Nes03, Theorem 2.1.15] for instance).

From a high-level perspective, Nesterov acceleration iterates over several variables, alternating between gradient steps (always with respect to the gradient at y_k) and mixing steps, where the running value of a variable is replaced by a linear combination of the other variables. However, the precise way gradient and mixing steps are coupled is rather mysterious, and the success of the proof of Theorem 2.1 relies heavily on the detailed structure of the iterations. In the next section, we try to gain perspective on this structure by developing a continuized version of the acceleration.

2.4. Continuized version of Nesterov acceleration

This chapter uses several mathematical notions related to random processes. The following sections expose the results from heuristic considerations of those notions, rigorously defined in Section 2.A.

We argue that the accelerated iteration becomes more natural when considering two variables x_t, z_t indexed by a continuous time $t \geq 0$, that are continuously mixing and that take gradient steps at random times. More precisely, let $T_1, T_2, T_3, \dots \geq 0$ be random times such that $T_1, T_2 - T_1, T_3 - T_2, \dots$ are independent identically distributed (i.i.d.), of law exponential with rate 1 (any constant rate would do, we choose 1 to make the comparison with discrete time k straightforward). By convention, we choose that our stochastic processes $t \mapsto x_t, t \mapsto z_t$ are càdlàg almost surely, i.e., right continuous with well-defined left-limits x_{t-}, z_{t-} (Definition 2.A.5 in Appendix 2.A). Our dynamics are parametrized by functions $\gamma_t, \gamma'_t, \eta_t, \eta'_t$, $t \geq 0$. At random times $T_1, T_2, \dots$, our sequences take gradient steps

$$x_{T_k} = x_{T_k-} - \gamma_{T_k} \nabla f(x_{T_k-}) \,, \tag{2.4}$$
$$z_{T_k} = z_{T_k-} - \gamma'_{T_k} \nabla f(x_{T_k-}) \,. \tag{2.5}$$

Because of the memoryless property of the exponential distribution, in a infinitesimal time interval $[t, t + \mathrm{d}t]$, the variables take gradients steps with probability $\mathrm{d}t$, independently of the past. Between these random times, the variables mix through a linear, translation-invariant, ordinary differential equation (ODE)

$$\mathrm{d}x_t = \eta_t (z_t - x_t) \mathrm{d}t \,, \tag{2.6}$$
$$\mathrm{d}z_t = \eta'_t (x_t - z_t) \mathrm{d}t \,. \tag{2.7}$$

Following the notation of stochastic calculus, we can write the process more compactly in terms of the Poisson point measure $\mathrm{d}N(t) = \sum_{k \geq 1} \delta_{T_k}(\mathrm{d}t)$, which has intensity the Lebesgue measure $\mathrm{d}t$,

$$\mathrm{d}x_t = \eta_t (z_t - x_t) \mathrm{d}t - \gamma_t \nabla f(x_t) \mathrm{d}N(t) \,, \tag{2.8}$$
$$\mathrm{d}z_t = \eta'_t (x_t - z_t) \mathrm{d}t - \gamma'_t \nabla f(x_t) \mathrm{d}N(t) \,. \tag{2.9}$$

Before giving convergence guarantees for such processes, let us digress quickly on why we can expect an iteration of this form to be mathematically appealing.

First, to a Markov chain indexed by a discrete time index k, one can associate the so-called *continuized* Markov chain, indexed by a continuous time t, that makes transition with the same Markov kernel, but at random times, with independent exponential time intervals [AF02]. Following this terminology, we refer to our acceleration (2.8)-(2.9) as the continuized acceleration. The continuized Markov chain is appreciated for its continuous time parameter t, while keeping many properties of the original Markov chain; similarly the continuized acceleration is arguably simpler to analyze, while performing similarly to Nesterov acceleration.

Second, it can also be compared with coordinate gradient descent methods, that are easier to analyze when coordinates are selected randomly rather than in an ordered way [Wri15]. Similarly, the continuized acceleration is simpler to analyze because the gradient steps (2.4)-(2.5) and the mixing steps (2.6)-(2.7) alternate randomly, due to the randomness of T_k, $k \geq 0$.

In analogy with Theorem 2.1, we give choices of parameters that lead to accelerated convergence rates, in the convex case (1) and in the strongly convex case (2). Convergence is analyzed as a function of t. As $\mathrm{d}N(t)$ is a Poisson point process with rate 1, t is the expected number of gradient steps done by the algorithm. Thus t is analoguous to k in Theorem 2.1.

In the theorem below, $\mathbb{E}$ denotes the expectation with respect to the Poisson point process $\mathrm{d}N(t)$, the only source of randomness.

Theorem 2.2 (Convergence of continuized Nesterov acceleration)**.** *The continuized Nesterov acceleration satisfies the following two points.*

1. *Choose the parameters* $\eta_t = \frac{2}{t}, \eta'_t = 0, \gamma_t = \frac{1}{L}, \gamma'_t = \frac{t}{2L}$. *Then*
$$\mathbb{E}f(x_t) - f(x_*) \leq \frac{2L\|z_0 - x_*\|^2}{t^2}\,.$$
2. *Assume further that* f *is* μ*-strongly convex,* $\mu > 0$. *Choose the constant parameters* $\eta_t = \eta'_t \equiv \sqrt{\frac{\mu}{L}}$, $\gamma_t \equiv \frac{1}{L}$, $\gamma'_t \equiv \frac{1}{\sqrt{\mu L}}$. *Then*
$$\mathbb{E}f(x_t) - f(x_*) \leq \left(f(x_0) - f(x_*) + \frac{\mu}{2}\|z_0 - x_*\|^2\right)\exp\left(-\sqrt{\frac{\mu}{L}}t\right)\,.$$

We give an elementary sketch of proof below and a complete proof in Appendix 2.B.1. Many authors have proposed continuous-time versions of Nesterov acceleration using differential calculus, see the numerous references in the introduction. For instance, in [SBC14], an ODE is obtained from Nesterov acceleration by taking the joint asymptotic where the stepsizes vanish and the number of iterates is rescaled. The resulting ODE must be discretized to be implemented; choosing the right discretization is not straightforward as it introduces stability and approximation errors that must be controlled [ZMSJ18, SDSJ19, SSZ20].

On the contrary, our continuous time process (2.8)-(2.9) does not correspond to a limit where the stepsizes vanish.

Remark 2.4.1. *A similar Markovian structure can be obtained in a discrete setting by flipping* i.i.d. *coins to trigger gradient steps. By denoting* $p > 0$ *the probability to trigger a gradient step when flipping a coin,* (i) $p = 1$ *gives the classical setting, and* (ii) $p \to 0$ *while renormalizing time gives our continuized framework. In fact, this setting with updates triggered randomly is an interpolation between the classical and continuized frameworks, and consists in replacing exponential random variables by geometric random variables of parameter* p *for the waiting-time between updates. We thus believe the convergence guarantees described here and in the following can be adapted for this discrete scheme.*

Sketch of proof for Theorem 2.2. A complete and rigorous proof is given in Appendix 2.B.1. Here, we only provide the heuristic of the main lines of the proof. The proof is similar to the one of Nesterov acceleration: we prove that for some choices of parameters $\eta_t, \eta'_t, \gamma_t, \gamma'_t$, $t \geq 0$, and for some functions A_t, B_t, $t \geq 0$,
$$\phi_t = A_t\left(f(x_t) - f(x_*)\right) + \frac{B_t}{2}\|z_t - x_*\|^2$$
is a supermartingale. In particular, this implies that $\mathbb{E}\phi_t$ is a Lyapunov function, i.e., a non-increasing function of t.

To prove that ϕ_t is a supermartingale, it is sufficient to prove that for all infinitesimal time intervals $[t, t + \mathrm{d}t]$, $\mathbb{E}_t\phi_{t+\mathrm{d}t} \leq \phi_t$, where $\mathbb{E}_t$ denotes the conditional expectation knowing all the past of the Poisson process up to time t. Thus we would like to compute the first order variation of $\mathbb{E}_t\phi_{t+\mathrm{d}t}$. This implies computing the first order variation of $\mathbb{E}_t f(x_{t+\mathrm{d}t})$.

From (2.8), we see that $f(x_t)$ evolves for two reasons between t and $t + \mathrm{d}t$:

- x_t follows the linear ODE (2.6), which results in the infinitesimal variation $f(x_t) \to f(x_t) + \eta_t\langle\nabla f(x_t), z_t - x_t\rangle\mathrm{d}t$, and

- with probability $\mathrm{d}t$, x_t takes a gradient step, which results in a macroscopic variation $f(x_t) \to f\left(x_t - \gamma_t \nabla f(x_t)\right)$.

Combining both variations, we obtain that

$$\mathbb{E}_t f(x_{t+\mathrm{d}t}) \approx f(x_t) + \eta_t \langle \nabla f(x_t), z_t - x_t \rangle \mathrm{d}t + \mathrm{d}t \left(f\left(x_t - \gamma_t \nabla f(x_t)\right) - f(x_t) \right) ,$$

where the $\mathrm{d}t$ in the second term corresponds to the probability that a gradient step happens; note that the latter event is independent of the past up to time t.

A similar computation can be done for $\mathbb{E}_t \|z_t - x_*\|^2$. Putting things together, we obtain

$$\begin{aligned}\mathbb{E}_t \phi_{t+\mathrm{d}t} - \phi_t \approx \mathrm{d}t \Bigg(& \frac{\mathrm{d}A_t}{\mathrm{d}t}(f(x_t) - f(x_*)) + A_t \eta_t \langle \nabla f(x_t), z_t - x_t \rangle \\ & - A_t \left(f\left(x_t - \gamma_t \nabla f(x_t)\right) - f(x_t) \right) + \frac{\mathrm{d}B_t}{\mathrm{d}t} \frac{1}{2} \|z_t - x_*\|^2 \\ & + B_t \eta_t' \langle z_t - x_*, x_t - z_t \rangle + \frac{B_t}{2} \left(\|z_t - \gamma_t' \nabla f(x_t) - x_*\|^2 - \|z_t - x_*\|^2 \right) \Bigg) .\end{aligned}$$

Using convexity and strong convexity inequalities, and a few computations, we obtain the following upper bound:

$$\begin{aligned}\mathbb{E}_t \phi_{t+\mathrm{d}t} - \phi_t \lesssim \mathrm{d}t \Bigg(& \left(\frac{\mathrm{d}A_t}{\mathrm{d}t} - A_t \eta_t \right) \langle \nabla f(x_t), x_t - x_* \rangle + \left(\frac{\mathrm{d}B_t}{\mathrm{d}t} - B_t \eta_t' \right) \frac{1}{2} \|z_t - x_*\|^2 \\ & + (A_t \eta_t - B_t \gamma_t') \langle \nabla f(x_t), z_t - x_* \rangle + \left(B_t \eta_t' - \frac{\mathrm{d}A_t}{\mathrm{d}t} \mu \right) \frac{1}{2} \|x_t - x_*\|^2 \\ & + \left(B_t \gamma_t'^2 - A_t \gamma_t \left(2 - L \gamma_t \right) \right) \frac{1}{2} \|\nabla f(x_t)\|^2 \Bigg) .\end{aligned}$$

We want this infinitesimal variation to be non-positive. Here, we choose the parameters so that $\gamma_t = 1/L$, and all prefactors in the above expression are zero. This gives some constraints on the choices of parameters. We show that only one degree of freedom is left: the choice of the function A_t, that must satisfy the ODE

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2} \left(\sqrt{A_t} \right) = \frac{\mu}{4L} \sqrt{A_t} ,$$

but whose initialization remains free. Once the initialization of the function A_t is chosen, this determines the full function A_t and, through the constraints, all parameters of the algorithm. As ϕ_t is a supermartingale (by design), a bound on the performance of the algorithm is given by

$$\mathbb{E} f(x_t) - f(x_*) \leq \frac{\mathbb{E}\phi_t}{A_t} \leq \frac{\phi_0}{A_t} .$$

The results presented in Theorem 2.2 correspond to one special choice of initialization for the function A_t.

In this sketch of proof, our derivation of the infinitesimal variation is intuitive and elementary; however it can be made more rigorous and concise—albeit more technical—using classical results from stochastic calculus, namely Proposition 2.A.2. This is our approach in Appendix 2.B.1.

□

2.5. Discrete implementation of the continuized acceleration with random parameters

In this section, we show that the continuized acceleration can be implemented exactly as a discrete algorithm. This contrasts with the discretization of ODEs that introduces discretization errors; here, we compute exactly

$$\tilde{x}_k := x_{T_k}\,, \qquad \tilde{y}_k := x_{T_{k+1}-}\,, \qquad \tilde{z}_k := z_{T_k}\,,$$

with the convention that $T_0 = 0$. The three sequences $\tilde{x}_k, \tilde{y}_k, \tilde{z}_k$, $k \geq 0$, satisfy a recurrence relation of the same structure as Nesterov acceleration, but with random weights. The resulting randomized discrete algorithm satisfies performance guarantees similar to those of Nesterov acceleration.

Theorem 2.3 (Discrete version of continuized acceleration)**.** *For any stochastic process of the form (2.8)-(2.9), we have*

$$\tilde{y}_k = \tilde{x}_k + \tau_k(\tilde{z}_k - \tilde{x}_k)\,, \tag{2.10}$$
$$\tilde{x}_{k+1} = \tilde{y}_k - \tilde{\gamma}_k \nabla f(\tilde{y}_k)\,, \tag{2.11}$$
$$\tilde{z}_{k+1} = \tilde{z}_k + \tau'_k(\tilde{y}_k - \tilde{z}_k) - \tilde{\gamma}'_k \nabla f(\tilde{y}_k)\,, \tag{2.12}$$

for some random parameters $\tau_k, \tau'_k, \tilde{\gamma}_k, \tilde{\gamma}'_k$ (that are functions of $T_k, T_{k+1}, \eta_t, \eta'_t, \gamma_t, \gamma'_t$).

1. *For the parameters of Theorem 2.2.(1), $\tau_k = 1 - \left(\frac{T_k}{T_{k+1}}\right)^2$, $\tau'_k = 0$, $\tilde{\gamma}_k = \frac{1}{L}$, and $\tilde{\gamma}'_k = \frac{T_k}{2L}$. Then*

$$\mathbb{E}\left[T_k^2\left(f(\tilde{x}_k) - f(x_*)\right)\right] \leq 2L\|z_0 - x_*\|^2\,.$$

2. *For the parameters of Theorem 2.2.(2), $\tau_k = \frac{1}{2}\left(1 - \exp\left(-2\sqrt{\frac{\mu}{L}}(T_{k+1} - T_k)\right)\right)$, $\tau'_k = \tanh\left(\sqrt{\frac{\mu}{L}}(T_{k+1} - T_k)\right)$, $\tilde{\gamma}_k = \frac{1}{L}$, and $\tilde{\gamma}'_k = \frac{1}{\sqrt{\mu L}}$. Then*

$$\mathbb{E}\left[\exp\left(\sqrt{\frac{\mu}{L}}T_k\right)\left(f(\tilde{x}_k) - f(x_*)\right)\right] \leq f(x_0) - f(x_*) + \frac{\mu}{2}\|z_0 - x_*\|^2\,.$$

The law of T_k is well known: it is the sum of k i.i.d. random variables of law exponential with rate 1; this is called an Erlang or Gamma distribution with shape parameter k and rate 1. One can use well-known properties of this law, such as its concentration around its expectation $\mathbb{E}T_k = k$, to derive corollaries of the bounds above. The performance guarantees are proved in Appendix 2.B.1.

2.6. Continuized Nesterov acceleration of stochastic gradient descent

We now investigate the design of continuized accelerations of stochastic gradient descent. We assume that we do not have direct access to the gradient $\nabla f(x)$ but to a random estimate $\nabla f(x, \xi)$, where $\xi \in \Xi$ is random of law $\mathcal{P}$. In the continuized framework, the randomness of the stochastic gradient and its time mix in a particularly convenient way. For similar reasons, Latz studied stochastic gradient descent as a gradient flow on a random function that is regenerated at a Poisson rate [Lat21]. However, this approach has the same shortcomings as the other approaches based on gradient flows: the subsequent discretization introduces non-trivial errors. We avoid this problem here.

We keep the algorithms of the same form, replacing gradients by stochastic gradients. Let $\xi_1, \xi_2, \ldots$ be i.i.d. random variables of law $\mathcal{P}$. We take stochastic gradient steps at the

random times $T_1, T_2, \dots,$

$$x_{T_k} = x_{T_k-} - \gamma_{T_k} \nabla f(x_{T_k-}, \xi_k)\,,$$
$$z_{T_k} = z_{T_k-} - \gamma'_{T_k} \nabla f(x_{T_k-}, \xi_k)\,.$$

Between these random times, the variables mix through the same ODE

$$\mathrm{d}x_t = \eta_t(z_t - x_t)\mathrm{d}t\,,$$
$$\mathrm{d}z_t = \eta'_t(x_t - z_t)\mathrm{d}t\,.$$

This can be written more compactly in terms of the Poisson point measure $\mathrm{d}N(t,\xi) = \sum_{k\geq 1} \delta_{(T_k,\xi_k)}(\mathrm{d}t, \mathrm{d}\xi)$ on $\mathbb{R}_{\geq 0} \times \Xi$, which has intensity $\mathrm{d}t \otimes \mathcal{P}$,

$$\mathrm{d}x_t = \eta_t(z_t - x_t)\mathrm{d}t - \gamma_t \int_\Xi \nabla f(x_t, \xi)\mathrm{d}N(t,\xi)\,, \tag{2.13}$$

$$\mathrm{d}z_t = \eta'_t(x_t - z_t)\mathrm{d}t - \gamma'_t \int_\Xi \nabla f(x_t, \xi)\mathrm{d}N(t,\xi)\,. \tag{2.14}$$

Here, the discussion depends on the properties satisfied by the stochastic gradients $\nabla f(x,\xi)$.

We now focus on functions f is of the following form, typical to least-squares supervised learning:

$$\forall x \in \mathbb{R}^d,\ f(x) = \mathbb{E}_{(a,b)\sim\mathcal{P}}\left[\frac{1}{2}\left(b - \langle x, a\rangle\right)^2\right], \tag{2.15}$$

where $\xi = (a,b) \in \mathbb{R}^d \times \mathbb{R}$ is random of law $\mathcal{P}$. We assume that our *stochastic first order oracle* is the gradient of one realization of the expectation, namely,

$$\nabla f(x,\xi) = -(b - \langle x, a\rangle)a\,, \qquad \xi = (a,b)\,.$$

We investigate *noiseless*—or purely multiplicative—stochastic gradients, in the sense that almost surely, for $\xi = (a,b) \sim \mathcal{P}$:

$$b = \langle x_*, a\rangle,\ \text{so that } \nabla f(x_*, \xi) = 0\,. \tag{2.16}$$

Noiseless stochastic gradients are relevant in several situations, such as coordinate gradient descent with randomly sampled coordinates [TY09, Nes12, Wri15] (where $\nabla f(x,\xi) = m\langle \nabla f(x), e_i\rangle e_i$ with i uniformly random in $\{1,\dots,d\}$), over-parameterized regime for least squares regression [VBS19], function interpolation and gossip algorithms [BBG20c].

For a symmetric non-negative matrix A and a vector x, we denote $\|x\|_A^2 = x^\top A x$. Let $H = \mathbb{E}[aa^\top]$ be the Hessian of f. Let R^2 be the smallest positive real number such that:

$$\mathbb{E}\left[\|a\|^2 aa^\top\right] \preccurlyeq R^2 H\,. \tag{2.17}$$

Further, similarly to [JKK+18], we define the statistical condition number of the problem as the smallest $\tilde{\kappa} > 0$ such that:

$$\mathbb{E}\left[\|a\|_{H^{-1}}^2 aa^\top\right] \preccurlyeq \tilde{\kappa} H\,. \tag{2.18}$$

Theorem 2.4 (Continuized acceleration with pure multiplicative noise)**.** *Assume that* (2.16), (2.17) *and* (2.18) *hold true. Then the continuized acceleration satisfies the following.*

1. *Choose the parameters* $\eta_t = \frac{2}{t}, \eta'_t = 0, \gamma_t = \frac{1}{R^2}, \gamma'_t = \frac{t}{2R^2\tilde{\kappa}}$. *Then*

$$\frac{1}{2}\mathbb{E}\|x_t - x_*\|^2 \leq \frac{R^2\tilde{\kappa}\|z_0 - x_*\|_{H^{-1}}^2}{t^2}\,.$$

2. *Assume further that f is μ-strongly convex, i.e., all eigenvalues of H are greater or equal to μ, where $\mu > 0$. The condition number of f is then defined as $\kappa = R^2/\mu$. For the parameters $\eta_t = \eta'_t = \frac{1}{\sqrt{\kappa\tilde{\kappa}}}$, $\gamma_t = \frac{1}{R^2}$ and $\gamma'_t = \frac{1}{R^2}\sqrt{\frac{\kappa}{\tilde{\kappa}}}$, we have:*

$$\frac{1}{2}\mathbb{E}\|x_t - x_*\|^2 \leq \left(\frac{1}{2}\|x_0 - x_*\|^2 + \frac{\mu}{2}\|z_0 - x_*\|^2_{H^{-1}}\right)\exp\left(-\frac{t}{\sqrt{\kappa\tilde{\kappa}}}\right).$$

In the strongly convex case, the benefits of this acceleration are similar to those of [JKK+18] with classical discrete iterates: while stochastic gradient descent with stepsize $1/R^2$ is easily shown to achieve an exponential rate of convergence $1/\kappa$, the acceleration enjoys a rate of convergence of $1/\sqrt{\kappa\tilde{\kappa}}$. Note that from the definitions, $\tilde{\kappa} \leq \kappa$, thus the acceleration performs as least as well as the naive algorithm. However, depending on the distribution of a, the improvement might either be significant or null. We refer the reader to the rich discussion in [JKK+18] which provides insights on the interpretation of $\tilde{\kappa}$ and on the possibility to accelerate. Below, we provide a complementary perspective on the statistical condition number in the context of gossip algorithms, where it can be interpreted in terms of effective resistances of graphs.

Albeit more restrictive in terms of assumptions, our analysis is much simpler than that of [JKK+18], as it relies on a standard Lyapunov function, similar to that of the continuized acceleration (Theorem 2.2). In Appendix 2.D, we use the same analysis framework to prove convergence of accelerated coordinate descent, which is another noiseless stochastic method.

2.7. Accelerating Randomized Gossip

The continuized framework allows designing accelerated decentralized algorithms requiring synchronized clocks, but no synchronization of the communications. In this section, we illustrate this statement in the simple case of gossip algorithms; the more general case of decentralized optimization is discussed in the next section.

Let $G = (\mathcal{V}, \mathcal{E})$ a connected graph representing a communication network of agents. Each agent $v \in \mathcal{V}$ is assigned a real number $x_0(v) \in \mathbb{R}$. The goal of the averaging (or gossip) problem is to design an iterative procedure allowing each agent of the network to know the average $\bar{x} = \frac{1}{m}\sum_{v\in\mathcal{V}} x_0(v)$ using only local communications, *i.e.*, communications between adjacent agents in the network.

We formalize the communication model of randomized gossip [BGPS06]. Time t is indexed continuously in $\mathbb{R}_{\geq 0}$. We generate a Poisson point measure $\mathrm{d}N(t,e) = \sum_{k\geq 1}\delta_{(T_k,\{v_k,w_k\})}$ with intensity measure $\mathrm{d}t \otimes \mathcal{P}$, where $\mathrm{d}t$ is the Lebesgue measure on $\mathbb{R}_{\geq 0}$ and $\mathcal{P} = (\mathcal{P}_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$ is a probability measure on the set $\mathcal{E}$ of edges. For $k \geq 0$, T_k is a time at which edge $\{v_k, w_k\}$ is *activated*: adjacent nodes v_k and w_k can communicate and perform a pairwise update. The Poisson point measure assumption implies that edges are activated independently of one another and from the past: the activation times of edge $\{v, w\}$ form a Poisson point process of intensity $\mathcal{P}_{\{v,w\}}$.

To solve the gossip problem, [BGPS06] proposed the following naive strategy: each agent $v \in \mathcal{V}$ keeps a local estimate $x_t(v)$ of the average and, upon activation of edge $\{v_k, w_k\}$ at time $T_k \in \mathbb{R}_{\geq 0}$, the activated nodes v_k, w_k average their current estimates

$$x_{T_k}(v_k),\, x_{T_k}(w_k) \quad \longleftarrow \quad \frac{x_{T_k-}(v_k) + x_{T_k-}(w_k)}{2}.$$

In this section, we accelerate this naive procedure. Our strategy is to apply Section 2.6 as follows. Consider the energy function

$$f(x) = \sum_{\{v,w\}\in\mathcal{E}} \frac{\mathcal{P}_{\{v,w\}}}{2}(x(v) - x(w))^2, \qquad x = (x(v))_{v\in\mathcal{V}}. \tag{2.19}$$

This function is convex, smooth, and writes in the form (2.15):

$$f(x) = \mathbb{E}_{\{v,w\}\sim\mathcal{P}}\left[\frac{1}{2}\left\langle x, a_{\{v,w\}}\right\rangle^2\right], \tag{2.20}$$

where $a_{\{v,w\}} = e_v - e_w$ and $(e_v)_{v\in\mathcal{V}}$ forms the canonical basis of $\mathbb{R}^{\mathcal{V}}$. As in Section 2.6, a stochastic gradient of f is obtained by taking the gradient of one realization of the expectation, namely:

$$\nabla f(x,\{v,w\}) = \langle x, a_{\{v,w\}}\rangle a_{\{v,w\}} = \begin{cases} x(v) - x(w) & \text{at coordinate } v, \\ x(w) - x(v) & \text{at coordinate } w, \\ 0 & \text{at all other coordinates.} \end{cases} \tag{2.21}$$

As a consequence, a stochastic gradient step with stepsize $1/2$ corresponds to a local averaging alongside edge $\{v,w\}$, where $\{v,w\}\sim\mathcal{P}$. More generally, the randomized gossip algorithm as described by [BGPS06] is the stochastic gradient descent:

$$\mathrm{d}x_t = -\frac{1}{2}\int_{\mathbb{R}_{\geq 0}\times\mathcal{E}} \nabla f(x_t,\{v,w\})\mathrm{d}N(t,\{v,w\})\,. \tag{2.22}$$

Using Section 2.6, we can accelerate this algorithm if we know the strong convexity parameter of f and the constants R^2 and $\tilde{\kappa}$ as defined in (2.17) and (2.18) respectively. These constants can be intepreted as graph-related quantities here.

Definition 2.7.1 (Graph-related quantities)**.** *The Laplacian matrix $\mathcal{L}\in\mathbb{R}^{\mathcal{V}\times\mathcal{V}}$ of graph G with weights $(\mathcal{P}_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$ on the edges is the matrix with entries $\mathcal{L}_{v,w} = -\mathcal{P}_{\{v,w\}}$ if $\{v,w\}\in\mathcal{E}$, $\mathcal{L}_{v,v} = \sum_{w\sim v}\mathcal{P}_{\{v,w\}}$, and $\mathcal{L}_{v,w} = 0$ if $\{v,w\}\notin\mathcal{E}$. We denote μ_{gossip} the second smallest eigenvalue of $\mathcal{L}$, corresponding to its smallest positive eigenvalue. For $\{v,w\}\in\mathcal{E}$, let $R_{\text{eff}}(v,w) = (e(v)-e(w))^\top\mathcal{L}^{-1}(e(v)-e(w))$ be the effective resistance of edge $\{v,w\}$, and $R_{\max} = \max_{\{v,w\}\in\mathcal{E}} R_{\text{eff}}(v,w)$ be the maximal resistance in the graph.*

The function f is quadratic with Hessian $\mathcal{L}$, and strongly convex with parameter μ_{gossip} on the hyperplane $F = \{x\in\mathbb{R}^{\mathcal{V}} : \sum_{v\in\mathcal{V}} x(v) = \bar{x}\}$; hence we use the (perhaps abusive) notation μ_{gossip} throughout. Moreover, the conditions (2.17) and (2.18) are satisfied with $R^2 = 2$, $\tilde{\kappa} = R_{\max}$.

These parameters being given, the accelerated stochastic gradient descent updates (2.13)-(2.14) can be instantiated as follows. Each agent $v\in\mathcal{V}$ keeps two local estimates $x_t(v), z_t(v)$ of $\bar{x}$, initialized at $x_0(v)$. Upon activation of edge $\{v_k,w_k\}$ at time T_k,

$$\begin{aligned} x_{T_k}(v_k) = x_{T_k}(w_k) &= \frac{x_{T_k-}(v_k) + x_{T_k-}(w_k)}{2}\,, \\ z_{T_k}(v_k) &= z_{T_k-}(v_k) + \frac{1}{\sqrt{2\mu_{\text{gossip}}R_{\max}}}(x_{T_k-}(w_k) - x_{T_k-}(v_k))\,, \\ z_{T_k}(w_k) &= z_{T_k-}(w_k) + \frac{1}{\sqrt{2\mu_{\text{gossip}}R_{\max}}}(x_{T_k-}(v_k) - x_{T_k-}(w_k))\,. \end{aligned}$$

Between these updates, $x_t(v)$ and $z_t(v)$ locally mix at all nodes $v\in\mathcal{V}$, according to the coupled ODE:

$$\begin{aligned} \mathrm{d}x_t(v) &= \sqrt{\frac{2\mu_{\text{gossip}}}{R_{\max}}}(z_t(v) - x_t(v))\mathrm{d}t, \\ \mathrm{d}z_t(v) &= \sqrt{\frac{2\mu_{\text{gossip}}}{R_{\max}}}(x_t(v) - z_t(v))\mathrm{d}t. \end{aligned}$$

This algorithm is *asynchronous* in the sense that it does not require global synchronous operations: the mixing of local variables does not require any synchronization since parameter $t \in \mathbb{R}_{\geq 0}$ is available at all nodes independently from the number of past updates, while a local pairwise update between adjacent nodes v and w only requires a local synchronization.

Theorem 2.5 (Accelerated randomized gossip)**.** *Let $(x_t(v))_{v\in\mathcal{V},t\geq 0}$ be generated with accelerated randomized gossip. For any $t \in \mathbb{R}_{\geq 0}$:*

$$\sum_{v\in\mathcal{V}} \frac{1}{2}\mathbb{E}\left[\left(x_t(v)-\bar{x}\right)^2\right] \leq 2\left(\sum_{v\in\mathcal{V}} \frac{1}{2}\left(x_0(v)-\bar{x}\right)^2\right)\exp\left(-\sqrt{\frac{\mu_{\text{gossip}}}{2R_{\max}}}t\right).$$

Let $\theta_{\text{ARG}} = \sqrt{\frac{\mu_{\text{gossip}}}{2R_{\max}}}$ be the rate of convergence of accelerated randomized gossip, and $\theta_{\text{RG}} = \mu_{\text{gossip}}$ be the rate of convergence of randomized gossip [BGPS06]. We have $\theta_{\text{ARG}} \geq \theta_{\text{RG}}/\sqrt{2}$. Let us exhibit scenarios over which accelerated randomized gossip gains several orders of magnitude. Denoting $\mathcal{P}_{\min} = \min_{\{v,w\}\in\mathcal{E}} \mathcal{P}_{\{v,w\}}$, [ESV+11] ensures that for $\{v,w\} \in \mathcal{E}$, $\mathcal{P}_{\min} R_{\text{eff}}(v,w) \leq 1$, so that $R_{\max} \leq \mathcal{P}_{\min}^{-1}$.

Corollary 2.7.1 (Comparison with randomized gossip)**.** *Accelerated randomized gossip achieves a rate satisfying:*

$$\sqrt{\frac{\theta_{\text{RG}}\mathcal{P}_{\min}}{2}} \leq \theta_{\text{ARG}}.$$

Assume furthermore that there exist some constants $c > 0$ such that for all $\{v,w\} \in \mathcal{E}$, $\mathcal{P}_{\{v,w\}} \leq c\mathcal{P}_{\min}$ and $d_v + d_w \leq 2d$. Then, with $C = 1/\sqrt{2cd}$:

$$C\sqrt{\frac{\theta_{\text{RG}}}{|\mathcal{V}|}} \leq \theta_{\text{ARG}}.$$

Assume now for simplicity that the Poisson intensities $\mathcal{P}_{\{v,w\}}$ are all equal to $1/|\mathcal{E}|$. Denoting $|\mathcal{V}| = m$, on the cyclic and the line graph, this gives us $\theta_{\text{ARG}} = \Omega(1/m^2)$ while $\theta_{\text{RG}} \asymp 1/m^3$. On a d-dimensional grid, we have $\theta_{\text{ARG}} = \Omega(1/m^{1+1/d})$ and $\theta_{\text{RG}} \asymp 1/m^{1+2/d}$. However, on graphs with unbounded degrees, no improvements are observed. We thus recover the same rates as [DSW08] for the graphs they study, but generalized to any network.

2.8. Accelerating Asynchronous Decentralized Optimization

Our continuized framework for accelerating randomized gossip can be extended to the more general problem of decentralized optimization: each node v in the network G previously defined holds a function $f_v : \mathbb{R}^d \to \mathbb{R}$, μ-strongly convex and L-smooth. Nodes of the network collaborate to solve:

$$\min_{x\in\mathbb{R}^d}\left\{f(x) = \frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}} f_v(x)\right\}. \tag{2.23}$$

As in gossip averaging, only local communications are allowed. Quantities related to f_v can only be computed at node v. In the case of empirical risk minimization, f_v represents the empirical risk related to node v's local data. Setting $f_v(x) = \frac{1}{2}\|x - x_0(v)\|^2$ leads to the averaging problem previously described. Similarly to Section 2.7, time is indexed continuously by t in $\mathbb{R}_{\geq 0}$, and communications are ruled by the same Poisson point measure $\mathrm{d}N(t,e) = \sum_{k\geq 1}\delta_{(T_k,\{v_k,w_k\})}$ on $\mathbb{R}_{\geq 0}\times\mathcal{E}$.

We first rewrite Problem (2.23) as:

$$\min_{X\in\mathbb{R}^{|\mathcal{V}|\times d},\ X_u = X_v\ \forall\{u,v\}\in\mathcal{E}}\left\{F(X) = \frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}} f_v(X_v)\right\}, \tag{2.24}$$

where $X_v \in \mathbb{R}^d$ corresponds to the local parameter of node v, and the equality constraints ensures equivalence between (2.23) and (2.24). The constraints are linear and can be expressed in matrix form as:

$$A^\top X = 0, \tag{2.25}$$

with $A \in \mathbb{R}^{\mathcal{E}\times\mathcal{V}}$ such that $\ker(A^\top) = \mathrm{Span}(1, ..., 1)$ the constant vector. The natural choice for matrix A is to choose a square root of the Laplacian matrix of graph G. For $(e_v)_{v\in\mathcal{V}}$ and $(e_{\{v,w\}})_{\{v,w\}\in\mathbb{E}}$ the canonical bases of $\mathbb{R}^{\mathcal{V}}$ and $\mathbb{R}^{\mathcal{E}}$, A is thus that for any $\{v, w\} \in \mathcal{E}$:

$$Ae_{\{v,w\}} = \sqrt{\mathcal{P}_{\{v,w\}}}(e_v - e_w).$$

Matrix A then satisfies $AA^\top = \mathcal{L}$ the Laplacian matrix of graph G with weights $\mathcal{P}_{\{v,w\}}$. Indeed, if $W_{\{v,w\}} = \mathcal{P}_{\{v,w\}}(e_v - e_w)(e_v - e_w)^\top$ corresponds to the gossip matrix for edge $\{v, w\}$, A is such that:

$$AA^\top = \sum_{\{v,w\}\in\mathcal{E}} W_{\{v,w\}} = \mathcal{L}. \tag{2.26}$$

Then, introducing Lagrange multipliers λ, we obtain through Lagrangian duality that Problem (2.23) is equivalent to:

$$\max_{\lambda\in\mathbb{R}^{\mathcal{E}\times d}} -F^*(A\lambda), \tag{2.27}$$

with F^* the convex conjugate of F. Following the approach of [HBM18], we then apply Accelerated Coordinate Descent to this dual problem. Yet, we use the *continuized* version of Theorem 2.8, which allows us to remove the global iterations counter on which previous approaches rely. We see that Problem (2.27) has exactly the right form to apply Theorem 2.8, leading to the following dual iterations:

$$\begin{aligned}
\mathrm{d}\lambda_t^{(y)} &= \eta_t(\lambda_t^{(z)} - \lambda_t^{(y)})\mathrm{d}t - \gamma_t \int_{\mathbb{R}_{\geq 0}\times\mathcal{E}} \frac{R_{\{v,w\}}}{\mathcal{P}^2_{\{v,w\}}} e_{\{v,w\}}e_{\{v,w\}}^\top A^\top \nabla F^*(A\lambda_t^{(y)})\mathrm{d}N(t, \{v, w\})\,, \\
\mathrm{d}\lambda_t^{(z)} &= \eta'_t(\lambda_t^{(y)} - \lambda_t^{(z)})\mathrm{d}t - \gamma'_t \int_{\mathbb{R}_{\geq 0}\times\mathcal{E}} \frac{1}{\mathcal{P}_{\{v,w\}}} e_{\{v,w\}}e_{\{v,w\}}^\top A^\top \nabla F^*(A\lambda_t^{(y)})\mathrm{d}N(t, \{v, w\})\,,
\end{aligned} \tag{2.28}$$

where $P = A^\dagger A$ with $A^\dagger$ is the pseudo-inverse of A, $R_{\{v,w\}} = e_{\{v,w\}}^\top A^\dagger A e_{\{v,w\}}$. Now, we multiply these iterations by A on the left (which is standard), and we rewrite them with the following iterates:

$$y_t = A\lambda_t^{(y)}, \qquad z_t = A\lambda_t^{(z)}. \tag{2.29}$$

Note that $y_t, z_t \in \mathbb{R}^{|\mathcal{V}|\times d}$, and are thus variables associated with *nodes* of the graph.

$$\begin{aligned}
\mathrm{d}y_t &= \eta_t(z_t - y_t)\mathrm{d}t - \gamma_t \int_{\mathbb{R}_{\geq 0}\times\mathcal{E}} \frac{R_{\{v,w\}}}{\mathcal{P}^2_{\{v,w\}}} W_{\{v,w\}} \nabla F^*(y_t)\mathrm{d}N(t, \{v, w\})\,, \\
\mathrm{d}z_t &= \eta'_t(y_t - z_t)\mathrm{d}t - \gamma'_t \int_{\mathbb{R}_{\geq 0}\times\mathcal{E}} \frac{1}{\mathcal{P}_{\{v,w\}}} W_{\{v,w\}} \nabla F^*(y_t)\mathrm{d}N(t, \{v, w\})\,,
\end{aligned} \tag{2.30}$$

where we recall that $W_{\{v,w\}} = \mathcal{P}_{\{v,w\}}(e_v - e_w)(e_v - e_w)^\top$ corresponds to the gossip matrix for edge $\{v, w\}$. Besides, the dual gradients $\nabla F^*(y_t)$ are such that $e_v^\top \nabla F^*(y_t) = \nabla f_v^*(e_v^\top y_t)$, and so each component can be computed locally at node v.

In summary, the distributed decentralized algorithm writes as follows. Upon activation

of edge $\{v_k, w_k\}$ at time T_k,

$$
\begin{aligned}
G_{\{v_k,w_k\}}(T_k) &= W_{\{v_k,w_k\}}\left[\nabla f^*((y_{T_k^-})_{v_k}) - \nabla f^*((y_{T_k^-})_{w_k})\right] \\
y_{T_k}(v_k) &= y_{T_k^-}(v_k) - \gamma_t \frac{R_{\{v_k,w_k\}}}{\mathcal{P}^2_{\{v_k,w_k\}}} G_{\{v_k,w_k\}}(T_k)\,, \\
y_{T_k}(w_k) &= y_{T_k^-}(w_k) + \gamma_t \frac{R_{\{v_k,w_k\}}}{\mathcal{P}^2_{\{v_k,w_k\}}} G_{\{v_k,w_k\}}(T_k)\,, \\
z_{T_k}(v_k) &= z_{T_k^-}(v_k) - \gamma'_t G_{\{v_k,w_k\}}(T_k)\,, \\
z_{T_k}(w_k) &= z_{T_k^-}(w_k) + \gamma'_t G_{\{v_k,w_k\}}(T_k)\,.
\end{aligned}
\tag{2.31}
$$

Between these updates, $y_t(v)$ and $z_t(v)$ locally mix at all nodes $v \in \mathcal{V}$, according to the coupled ODE:

$$
\begin{aligned}
\mathrm{d}y_t(v) &= \eta_t(z_t(v) - y_t(v))\mathrm{d}t, \\
\mathrm{d}z_t(v) &= \eta'_t(y_t(v) - z_t(v))\mathrm{d}t.
\end{aligned}
$$

This algorithm can be implemented with local computations and pairwise communications only, since an update along edge $\{v, w\}$ only involves the parameters and functions of nodes v and w. In order to fully describe this algorithm, we need to specify the various parameters. We do so, with the corresponding rate of convergence, in the following theorem.

Theorem 2.6 (Accelerated asynchronous decentralized optimization)**.** *Let*

$$
\theta'_{\mathrm{ARG}} = \sqrt{\mu_{\mathrm{gossip}} / \max_{\{v,w\}} \frac{R_{\{v,w\}}}{\mathcal{P}_{\{v,w\}}}}\,,
$$

where μ_{gossip} is the smallest non-zero eigenvalue of the Laplacian of the graph $\mathcal{G}$, and $\kappa = L/\mu$. Denoting $(x_t(v))_{v\in\mathcal{V}} = (\nabla f_v^*(z_t(v)))_{v\in\mathcal{V}}$ generated by the accelerated coordinate descent on the dual of Problem (2.23):

$$
\sum_{v\in\mathcal{V}} \frac{1}{2}\mathbb{E}\left[\|x_t(v) - x_*\|^2\right] \leq C\left(\sum_{v\in\mathcal{V}} \frac{1}{2}\|x_0(v) - x_*\|^2\right) \exp\left(-\frac{\theta'_{\mathrm{ARG}}}{\sqrt{\kappa}} t\right)\,,
$$

where $\kappa = \mu/L$ is an upper bound on the condition number of f, C is a constant that depends on the graph and κ, and θ'_{ARG} is the rate of convergence of accelerated randomized gossip on the graph G.

This algorithm thus features the same communication accelerationg as accelerated randomized gossip, and the same discussion thus applies. However, due to the methodology to derive the updates, gradients of the dual conjugates f_v^* are involved. These quantities are usually hard to compute, except in simple cases such as quadratic objectives, limiting the applicatbility of our continuized acceleration framework.

However, the acceleration scheme we developed can be used to accelerate decentralized SGD using *primal* local updates: this has been done in the works [NO23, NBO23], that coupled our accelerated randomized gossip with local SGD steps in the continuized framework.

2.9. Conclusion

In this work, we introduced a continuized version of Nesterov's accelerated gradients. In a nutshell, the method has two sequences of iterates which take gradient steps at random times.

In between gradient steps, the two sequences mix following a simple ordinary differential equation, whose parameters are picked to ensure good convergence properties of the method.

As compared to other continuous time models of Nesterov acceleration, a key feature of this approach is that the method can be implemented without any approximation, as the differential equation governing the mixing procedure has a simple analytical solution. A discretization of the continuized method corresponds to an accelerated gradient method with random parameters.

Continuization strategies were introduced in the context of Markov chains [AF02]. Here, they allow using acceleration mechanisms in asynchronous distributed optimization, where usually agents are not aware of the total number of iterations taken so far. This is showcased in the context of asynchronous gossip algorithms, and has further been studied by other authors [NO23, NBO23] to develop large-scale asynchronous decentralized algorithms.

Finally, the continuized Nesterov acceleration we introduced also proves to be the first acceleration of *quasar convex functions* [WW23], a more general class of functions than convex functions.

Appendix of Chapter 2

2.A. Stochastic calculus toolbox

In this appendix, we give a short introduction to the mathematical tools that we used in this chapter. For more details, the reader can consult the more rigorous monographs of [JS13, IW14, LG16].

2.A.1. Poisson point measures

We fix $\mathcal{P}$ a probability law on some space Ξ.

Definition 2.A.1. *A* (homogenous) Poisson point measure *on $\mathbb{R}_{\geq 0}\times\Xi$, with intensity $\nu(\mathrm{d}t,\mathrm{d}\xi) = \mathrm{d}t\otimes\mathrm{d}\mathcal{P}(\xi)$, is a random measure N on $\mathbb{R}_{\geq 0}\times\Xi$ such that*

- *For any disjoint measurable subsets A and B of $\mathbb{R}_{\geq 0}\times\Xi$, $N(A)$ and $N(B)$ are independent.*
- *For any measurable subset A of $\mathbb{R}_{\geq 0}\times\Xi$, $N(A)$ is a Poisson random variable with parameter $\nu(A)$. (If $\nu(A)=\infty$, $N(A)$ is equal to ∞ almost surely.)*

Proposition 2.A.1. *Let N be a Poisson point measure on $\mathbb{R}_{\geq 0}\times\Xi$ with intensity $\mathrm{d}t\otimes\mathrm{d}\mathcal{P}(\xi)$.*
There exists a decomposition $\mathrm{d}N(t,\xi) = \sum_{k\geq 1}\delta_{(T_k,\xi_k)}(\mathrm{d}t,\mathrm{d}\xi)$ on $\mathbb{R}_{\geq 0}\times\Xi$ where $0<T_1<T_2<T_3<\dots$ and $\xi_1,\xi_2,\xi_3,\dots\in\Xi$ satisfy:

- *$T_1, T_2-T_1, T_3-T_2,\dots$ are i.i.d. of law exponential with rate 1,*
- *$\xi_1,\xi_2,\xi_3,\dots$ are i.i.d. of law $\mathcal{P}$ and independent of the $T_1,T_2,T_3,\dots$.*

Definition 2.A.2. *Let N be a Poisson point measure on $\mathbb{R}_{\geq 0}\times\Xi$ with intensity $\mathrm{d}t\otimes\mathrm{d}\mathcal{P}(\xi)$. The* filtration *$\mathcal{F}_t$, $t\geq 0$,* generated by *N is defined by the formula*

$$\mathcal{F}_t = \sigma\left(N([0,s]\times A)\,,\ s\leq t, A\subset\Xi \text{ measurable}\right).$$

2.A.2. Martingales and supermartingales

Let $(\Omega,\mathcal{F},\mathbb{P})$ be a probability space and $\mathcal{F}_t$, $t\geq 0$, a filtration on this probability space.

Definition 2.A.3. *A random process $x_t\in\mathbb{R}^d$, $t\geq 0$, is* adapted *if for all $t\geq 0$, x_t is $\mathcal{F}_t$-measurable. An adapted process $x_t\in\mathbb{R}$, $t\geq 0$ is a* martingale *(resp.* supermartingale*) if for all $0\leq s\leq t$, $\mathbb{E}[x_t|\ \mathcal{F}_s] = x_s$ (resp. $\mathbb{E}[x_t|\ \mathcal{F}_s]\leq x_s$).*

Definition 2.A.4. *A random variable $T\in[0,\infty]$ is a* stopping time *if for all $t\geq 0$, $\{T\leq t\}\in\mathcal{F}_t$.*

Definition 2.A.5. *A function $x_t, t\geq 0$, is said to be* càdlàg *if it is right continuous and for every $t>0$, the limit $x_{t-} := \lim_{s\to t, s<t} x_s$ exists and is finite.*

Theorem 2.7 (Martingale stopping theorem)**.** *Let x_t, $t\geq 0$, be a martingale (resp. supermartingale) with càdlàg trajectories and uniformly integrable. Let T be a stopping time. Then $\mathbb{E}X_T = X_0$ (resp. $\mathbb{E}X_T\leq X_0$).*

2.A.3. Stochastic ordinary differential equation with Poisson jumps

The continuized processes are the composition of an ordinary differential equation and stochastic Poisson jumps. It is thus a piecewise-deterministic Markov process [Dav84, Dav18], a special case of stochastic models that do not include any diffusion term. The stochastic calculus of this class of processes is particularly intuitive: there is no Ito correction term as with diffusive processes.

We fix $\mathcal{P}$ a probability law on some space Ξ, N a Poisson point measure on $\mathbb{R}_{\geq 0} \times \Xi$ with intensity $\mathrm{d}t \otimes \mathrm{d}\mathcal{P}(\xi)$, and denote $\mathcal{F}_t$, $t \geq 0$, the filtration generated by N.

Definition 2.A.6. *Let $b : \mathbb{R}^d \to \mathbb{R}^d$ and $G : \mathbb{R}^d \times \Xi \to \mathbb{R}^d$ be two functions. An random process $x_t \in \mathbb{R}^d$, $t \geq 0$, is said to be a solution of the equation*

$$\mathrm{d}x_t = b(x_t)\mathrm{d}t + \int_{\Xi} G(x_t, \xi)\mathrm{d}N(t, \xi)$$

if it is adapted, càdlàg, and for all $t \geq 0$,

$$x_t = x_0 + \int_0^t b(x_s)\mathrm{d}s + \int_{[0,t]\times\Xi} G(x_{s-}, \xi)\mathrm{d}N(s, \xi)\,.$$

If we consider the decomposition $\mathrm{d}N(t, \xi) = \sum_{k\geq 1} \delta_{(T_k,\xi_k)}(\mathrm{d}t, \mathrm{d}\xi)$ given by Proposition 2.A.1, then

$$\int_{[0,t]\times\Xi} G(x_{s-}, \xi)\mathrm{d}N(s, \xi) = \sum_{k\geq 1} \mathbf{1}_{\{T_k \leq t\}} G(x_{T_k-}, \xi_k)\,.$$

Here, we consider only autonomous equations as b and G are a function of x_t, but not of t. However, there is no loss of generality, one can study time-dependent systems by studying the equation in the variable (t, x_t). This trick is used in Appendix 2.B.

Proposition 2.A.2. *Let $x_t \in \mathbb{R}^d$ be a solution of*

$$\mathrm{d}x_t = b(x_t)\mathrm{d}t + \int_{\Xi} G(x_t, \xi)\mathrm{d}N(t, \xi)$$

and $\varphi : \mathbb{R}^d \to \mathbb{R}$ be a smooth function. Then

$$\varphi(x_t) = \varphi(x_0) + \int_0^t \langle \nabla\varphi(x_s), b(x_s)\rangle \mathrm{d}s + \int_{[0,t]\times\Xi} \left(\varphi(x_{s-} + G(x_{s-}, \xi)) - \varphi(x_{s-})\right) \mathrm{d}N(s, \xi)\,.$$

Moreover, we have the decomposition

$$\begin{aligned}\int_{[0,t]\times\Xi} &\left(\varphi(x_{s-} + G(x_{s-}, \xi)) - \varphi(x_{s-})\right) \mathrm{d}N(s, \xi) \\ &= \int_0^t \int_{\Xi} \left(\varphi(x_s + G(x_s, \xi)) - \varphi(x_s)\right) \mathrm{d}t\mathrm{d}\mathcal{P}(\xi) + M_t\,,\end{aligned}$$

where $M_t = \int_{[0,t]\times\Xi} \left(\varphi(x_{s-} + G(x_{s-}, \xi)) - \varphi(x_{s-})\right) \left(\mathrm{d}N(s, \xi) - \mathrm{d}t\mathrm{d}\mathcal{P}(\xi)\right)$ is a martingale.

This proposition is an elementary calculus of variations formula: to compute the value of the observable $\varphi(x_t)$, one must sum the effects of the continuous part and of the Poisson jumps. Moreover, the integral with respect to the Poisson measure N becomes a martingale if the same integral with respect to its intensity measure $\mathrm{d}t \otimes \mathrm{d}\mathcal{P}(\xi)$ is removed.

2.B. Analysis of the continuized Nesterov acceleration

To encompass the proofs in the convex and in the strongly convex cases in a unified way, we assume f is μ-strongly convex, $\mu \geq 0$. If $\mu > 0$, this corresponds to assuming the μ-strong convexity in the usual sense; if $\mu = 0$, it means that we only assume the function to be convex. In other words, the proofs in the convex case can be obtained by taking $\mu = 0$ below. In this section, $\mathcal{F}_t$, $t \geq 0$, is the filtration associated to the Poisson point measure N.

2.B.1. Noiseless case: proofs of Theorem 2.2 and of the bounds of Theorem 2.3

In this section, we analyze the convergence of the continuized iteration (2.8)-(2.9), that we recall for the reader's convenience:

$$\begin{aligned} \mathrm{d}x_t &= \eta_t(z_t - x_t)\mathrm{d}t - \gamma_t \nabla f(x_t)\mathrm{d}N(t)\,, \\ \mathrm{d}z_t &= \eta'_t(x_t - z_t)\mathrm{d}t - \gamma'_t \nabla f(x_t)\mathrm{d}N(t)\,. \end{aligned}$$

The choices of parameters $\eta_t, \eta'_t, \gamma_t, \gamma'_t$, $t \geq 0$, and the corresponding convergence bounds follow naturally from the analysis. We seek sufficient conditions under which the function

$$\phi_t = A_t\left(f(x_t) - f_*\right) + \frac{B_t}{2}\|z_t - x_*\|^2$$

is a supermartingale.

The process $\bar{x}_t = (t, x_t, z_t)$ satisfies the equation

$$\mathrm{d}\bar{x}_t = b(\bar{x}_t)\mathrm{d}t + G(\bar{x}_t)\mathrm{d}N(t)\,, \qquad b(\bar{x}_t) = \begin{pmatrix} 1 \\ \eta_t(z_t - x_t) \\ \eta'_t(x_t - z_t) \end{pmatrix}\,, \qquad G(\bar{x}_t) = \begin{pmatrix} 0 \\ -\gamma_t \nabla f(x_t) \\ -\gamma'_t \nabla f(x_t) \end{pmatrix}\,.$$

We thus apply Proposition 2.A.2 to $\phi_t = \varphi(\bar{x}_t) = \varphi(t, x_t, z_t)$ where

$$\varphi(t, x, z) = A_t\left(f(x) - f(x_*)\right) + \frac{B_t}{2}\|z - x_*\|^2\,,$$

we obtain:

$$\phi_t = \phi_0 + \int_0^t \langle \nabla\varphi(\bar{x}_s), b(\bar{x}_s)\rangle \mathrm{d}s + \int_0^t \left(\varphi(\bar{x}_s + G(\bar{x}_s)) - \varphi(\bar{x}_s)\right)\mathrm{d}s + M_t\,,$$

where M_t is a martingale. Thus, to show that φ_t is a supermartingale, it is sufficient to show that the map $t \mapsto \int_0^t \langle \nabla\varphi(\bar{x}_s), b(\bar{x}_s)\rangle \mathrm{d}s + \int_0^t \left(\varphi(\bar{x}_s + G(\bar{x}_s)) - \varphi(\bar{x}_s)\right)\mathrm{d}s$ is non-increasing almost surely, i.e.,

$$I_t := \langle \nabla\varphi(\bar{x}_t), b(\bar{x}_t)\rangle + \varphi(\bar{x}_t + G(\bar{x}_t)) - \varphi(\bar{x}_t) \leq 0\,.$$

We now compute

$$\begin{aligned} \langle \nabla\varphi(\bar{x}_t), b(\bar{x}_t)\rangle &= \partial_t\varphi(\bar{x}_t) + \langle \partial_x\varphi(\bar{x}_t), \eta_t(z_t - x_t)\rangle + \langle \partial_z\varphi(\bar{x}_t), \eta'_t(x_t - z_t)\rangle \\ &= \frac{\mathrm{d}A_t}{\mathrm{d}t}\left(f(x_t) - f(x_*)\right) + \frac{\mathrm{d}B_t}{\mathrm{d}t}\frac{1}{2}\|z_t - x_*\|^2 + A_t\eta_t\langle \nabla f(x_t), z_t - x_t\rangle \\ &\quad + B_t\eta'_t\langle z_t - x_*, x_t - z_t\rangle\,. \end{aligned}$$

Here, we use that as f is μ-strongly convex,

$$f(x_t) - f(x_*) \leq \langle \nabla f(x_t), x_t - x_*\rangle - \frac{\mu}{2}\|x_t - x_*\|^2\,,$$

and the simple bound

$$\begin{aligned}\langle z_t - x_*, x_t - z_t\rangle &= \langle z_t - x_*, x_t - x_*\rangle - \|z_t - x_*\|^2\\ &\le \|z_t - x_*\|\|x_t - x_*\| - \|z_t - x_*\|^2\\ &\le \frac{1}{2}\left(\|z_t - x_*\|^2 + \|x_t - x_*\|^2\right) - \|z_t - x_*\|^2\\ &= \frac{1}{2}\left(\|x_t - x_*\|^2 - \|z_t - x_*\|^2\right).\end{aligned}$$

This gives

$$\langle \nabla\varphi(\bar{x}_t), b(\bar{x}_t)\rangle \le \left(\frac{\mathrm{d}A_t}{\mathrm{d}t} - A_t\eta_t\right)\langle \nabla f(x_t), x_t - x_*\rangle + \left(B_t\eta_t' - \frac{\mathrm{d}A_t}{\mathrm{d}t}\mu\right)\frac{1}{2}\|x_t - x_*\|^2 \quad (2.32)$$

$$+ \left(\frac{\mathrm{d}B_t}{\mathrm{d}t} - B_t\eta_t'\right)\frac{1}{2}\|z_t - x_*\|^2 + A_t\eta_t\langle \nabla f(x_t), z_t - x_*\rangle\,. \quad (2.33)$$

Further,

$$\begin{aligned}\varphi(\bar{x}_t + G(\bar{x}_t)) - \varphi(\bar{x}_t) = A_t\left(f(x_t - \gamma_t\nabla f(x_t)) - f(x_t)\right)\\ + \frac{B_t}{2}\left(\|(z_t - x_*) - \gamma_t'\nabla f(x_t)\|^2 - \|z_t - x_*\|^2\right).\end{aligned}$$

As f is L-smooth,

$$\begin{aligned}f(x_t - \gamma_t\nabla f(x_t)) - f(x_t) &\le \langle \nabla f(x_t), -\gamma_t\nabla f(x_t)\rangle + \frac{L}{2}\|\gamma_t\nabla f(x_t)\|^2\\ &= -\gamma_t\left(2 - L\gamma_t\right)\frac{1}{2}\|\nabla f(x_t)\|^2.\end{aligned}$$

This gives

$$\begin{aligned}&\varphi(\bar{x}_t + G(\bar{x}_t)) - \varphi(\bar{x}_t)\\ &\quad\le \left(B_t\gamma_t'^2 - A_t\gamma_t\left(2 - L\gamma_t\right)\right)\frac{1}{2}\|\nabla f(x_t)\|^2 - B_t\gamma_t'\langle \nabla f(x_t), z_t - x_*\rangle\,. \quad (2.34)\end{aligned}$$

Finally, combining (2.32)-(2.33) with (2.34), we obtain

$$\begin{aligned}I_t \le \left(\frac{\mathrm{d}A_t}{\mathrm{d}t} - A_t\eta_t\right)\langle \nabla f(x_t), x_t - x_*\rangle + \left(\frac{\mathrm{d}B_t}{\mathrm{d}t} - B_t\eta_t'\right)\frac{1}{2}\|z_t - x_*\|^2\\ + (A_t\eta_t - B_t\gamma_t')\langle \nabla f(x_t), z_t - x_*\rangle + \left(B_t\eta_t' - \frac{\mathrm{d}A_t}{\mathrm{d}t}\mu\right)\frac{1}{2}\|x_t - x_*\|^2\\ + \left(B_t\gamma_t'^2 - A_t\gamma_t\left(2 - L\gamma_t\right)\right)\frac{1}{2}\|\nabla f(x_t)\|^2.\end{aligned}$$

Remember that $I_t \le 0$ is a sufficient condition for ϕ_t to be a supermartingale. Here, we choose the parameters $\eta_t, \eta_t', \gamma_t, \gamma_t', t \ge 0$, so that all prefactors are 0. We start by taking $\gamma_t \equiv \frac{1}{L}$ (other choices $\gamma_t < \frac{2}{L}$ could be possible but would give similar results) and we want to satisfy

$$\frac{\mathrm{d}A_t}{\mathrm{d}t} = A_t\eta_t\,, \qquad \frac{\mathrm{d}B_t}{\mathrm{d}t} = B_t\eta_t' \qquad A_t\eta_t = B_t\gamma_t'\,, \qquad B_t\eta_t' = \frac{\mathrm{d}A_t}{\mathrm{d}t}\mu\,, \qquad B_t\gamma_t'^2 = \frac{A_t}{L}\,.$$

To satisfy the last equation, we choose

$$\gamma_t' = \sqrt{\frac{A_t}{LB_t}}\,. \quad (2.35)$$

To satisfy the third equation, we choose

$$\eta_t = \frac{B_t \gamma'_t}{A_t} = \sqrt{\frac{2B_t}{LA_t}}\,. \tag{2.36}$$

To satisfy the fourth equation, we choose

$$\eta'_t = \frac{\mathrm{d}A_t}{\mathrm{d}t}\frac{\mu}{B_t} = \frac{A_t \eta_t \mu}{B_t} = \mu\sqrt{\frac{A_t}{LB_t}}\,. \tag{2.37}$$

Having now all parameters $\eta_t, \eta'_t, \gamma_t, \gamma'_t$ constrained, we now have that ϕ_t is Lyapunov if

$$\frac{\mathrm{d}A_t}{\mathrm{d}t} = A_t\eta_t = \sqrt{\frac{A_tB_t}{L}}\,, \qquad\qquad \frac{\mathrm{d}B_t}{\mathrm{d}t} = B_t\eta'_t = \mu\sqrt{\frac{A_tB_t}{L}}\,.$$

This only leaves the choice of the initialization (A_0, B_0) as free: both the algorithm and the Lyapunov depend on it. (Actually, only the relative value A_0/B_0 matters.) Instead of solving the above system of two coupled non-linear ODEs, it is convenient to turn them into a single second-order linear ODE:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\sqrt{A_t}\right) = \frac{1}{2\sqrt{A_t}}\frac{\mathrm{d}A_t}{\mathrm{d}t} = \frac{1}{2}\sqrt{\frac{B_t}{L}}\,, \qquad \frac{\mathrm{d}}{\mathrm{d}t}\left(\sqrt{B_t}\right) = \frac{1}{2\sqrt{B_t}}\frac{\mathrm{d}B_t}{\mathrm{d}t} = \frac{\mu}{2}\sqrt{\frac{A_t}{L}}\,. \tag{2.38}$$

This can also be restated as

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\left(\sqrt{A_t}\right) = \frac{\mu}{4L}\sqrt{A_t}\,, \qquad\qquad \sqrt{B_t} = 2\sqrt{L}\frac{\mathrm{d}}{\mathrm{d}t}\left(\sqrt{A_t}\right)\,. \tag{2.39}$$

Proof of the first part (convex case) We now assume $\mu = 0$, and we choose the solution such that $A_0 = 0$ and $B_0 = 1$. From (2.38), we have $\frac{\mathrm{d}}{\mathrm{d}t}\left(\sqrt{B_t}\right) = 0$, thus $B_t \equiv 1$, and $\frac{\mathrm{d}}{\mathrm{d}t}\left(\sqrt{A_t}\right) = \frac{1}{2\sqrt{L}}$, thus $\sqrt{A_t} = \frac{t}{2\sqrt{L}}$. The parameters of the algorithm are given by (2.35)-(2.37): $\eta_t = \frac{2}{t}$, $\eta'_t = 0$, $\gamma'_t = \frac{t}{2L}$ (and we had chosen $\gamma_t = \frac{1}{L}$).

From the fact that ϕ_t is a supermartingale, we obtain that the associated algorithm satisfies

$$\mathbb{E}f(x_t) - f(x_*) \leq \frac{\mathbb{E}\phi_t}{A_t} \leq \frac{\phi_0}{A_t} = \frac{2L\|z_0 - x_*\|^2}{t^2}\,.$$

This proves the first part of Theorem 2.2.

Further, one can apply martingale stopping Theorem 2.7 to the supermartingale ϕ_t with the stopping time T_k to obtain

$$\mathbb{E}\left[A_{T_k}\left(f(\tilde{x}_k) - f(x_*)\right)\right] = \mathbb{E}\left[A_{T_k}\left(f(x_{T_k}) - f(x_*)\right)\right] \leq \mathbb{E}\phi_{T_k} \leq \phi_0 = \|z_0 - x_*\|^2\,.$$

This proves the formula of Theorem 2.3.1.

Proof of the second part (strongly convex case) We now assume $\mu > 0$. We consider the solution of (2.39) that is exponential:

$$\sqrt{A_t} = \sqrt{A_0}\exp\left(\frac{1}{2}\sqrt{\frac{\mu}{L}}t\right)\,, \qquad\qquad \sqrt{B_t} = \sqrt{A_0}\sqrt{\mu}\exp\left(\frac{1}{2}\sqrt{\frac{\mu}{L}}t\right)\,.$$

The parameters of the algorithm are given by (2.35)-(2.37): $\eta_t = \eta'_t = \sqrt{\frac{\mu}{L}}$, $\gamma'_t = \frac{1}{\sqrt{\mu L}}$ (and we had chosen $\gamma_t = \frac{1}{L}$).

From the fact that ϕ_t is a supermartingale, we obtain that the associated algorithm

satisfies

$$\mathbb{E}f(x_t) - f(x_*) \leq \frac{\mathbb{E}\phi_t}{A_t} \leq \frac{\phi_0}{A_t} = \frac{A_0(f(x_0) - f(x_*)) + A_0\frac{\mu}{2}\|z_0 - x_*\|^2}{A_t}$$
$$= \left(f(x_0) - f(x_*) + \frac{\mu}{2}\|z_0 - x_*\|^2\right)\exp\left(-\sqrt{\frac{\mu}{L}}t\right).$$

This proves the second part of Theorem 2.2. Similarly to above, one can also apply the martingale stopping theorem to prove the formula of Theorem 2.3.2.

Remark 2.B.1. *In the above derivation, in both the convex and strongly convex cases, we choose a particular solution of (2.39), while several solutions are possible. In the convex case, we make the choice $A_0 = 0$ to have a succinct bound that does not depend on $f(x_0) - f(x_*)$. More importantly, in the strongly convex case, we choose the solution that satisfies the relation $\sqrt{\mu}\sqrt{A_t} = \sqrt{B_t}$, which implies that $\eta_t, \eta'_t, \gamma'_t$, are constant functions of t, and $\eta_t = \eta'_t$. These conditions help solving in closed form the continuous part of the process*

$$\mathrm{d}x_t = \eta_t(z_t - x_t)\mathrm{d}t\,,$$
$$\mathrm{d}z_t = \eta'_t(x_t - z_t)\mathrm{d}t\,,$$

which is crucial if we want to have a discrete implementation of our method (for more details, see Theorem 2.3 and its proof). However, in the strongly convex case, considering other solutions would be interesting, for instance to have an algorithm converging to the convex one as $\mu \to 0$.

2.B.2. With Pure Multiplicative Noise: Proof of Theorem 2.4

The proof of this theorem mimics the proof of Theorem 2.2, with a slightly different Lyapunov function.

We recall that in Section 2.6, the function f is of the form:

$$\forall x \in \mathbb{R}^d, f(x) = \mathbb{E}\left[\frac{1}{2}(\langle a, x\rangle - b)^2\right],$$

where $\xi = (a, b) \in \mathbb{R}^d \times \mathbb{R}$ is of law $\mathcal{P}$. Thanks to the *noiseless assumption*, for $H = \mathbb{E}\left[aa^\top\right]$, we also have:

$$\forall x \in \mathbb{R}^d, f(x) = \frac{1}{2}\|x - x_*\|_H^2.$$

The Lyapunov function studied in the proof of Theorem 2.2 would then write as, for $t \in \mathbb{R}_{\geq 0}$:

$$\phi_t = \frac{A_t}{2}\|x_t - x_*\|_H^2 + \frac{B_t}{2}\|z_t - x_*\|^2.$$

An acceleration of stochastic gradient descent using this Lyapunov function has been done by [VBS19]. In order to have an analysis similar to Nesterov acceleration, the authors make a strong growth condition, which is too strong for many stochastic gradient problems and for our application to gossip algorithms. Instead, our analysis requires a bounded statistical condition number $\tilde{\kappa}$, and performs a shift in terms of dependency over H: $\|x - x_*\|_H^2$ becomes $\|x - x_*\|^2$, and $\|z_t - x_*\|^2$ becomes $\|z_t - x_*\|_{H^{-1}}^2$. The new Lyapunov function writes:

$$\phi_t = \frac{A_t}{2}\|x_t - x_*\|^2 + \frac{B_t}{2}\|z_t - x_*\|_{H^{-1}}^2.$$

As in Theorem 2.2, the proof consists in proving that for carefully chosen parameters, ϕ_t is

a supermatingale. The process $\bar{x}_t = (t, x_t, z_t)$ satisfies the equation

$$\begin{aligned}
\mathrm{d}\bar{x}_t &= b(\bar{x}_t)\mathrm{d}t + \int_\Xi G(\bar{x}_t, \xi)\mathrm{d}N(t,\xi)\,, \\
b(\bar{x}_t) &= \begin{pmatrix} 1 \\ \eta_t(z_t - x_t) \\ \eta_t'(x_t - z_t) \end{pmatrix}\,, \\
G(\bar{x}_t, \xi) &= \begin{pmatrix} 0 \\ -\gamma_t \nabla f(x_t, \xi) \\ -\gamma_t' \nabla f(x_t, \xi) \end{pmatrix}\,.
\end{aligned}$$

We apply Proposition 2.A.2 to $\phi_t = \varphi(\bar{x}_t) = \varphi(t, x_t, z_t)$ and obtain:

$$\phi_t = \phi_0 + \int_0^t I_s \mathrm{d}s + M_t\,,$$

where M_t is a martingale and

$$I_t = \langle \nabla\varphi(\bar{x}_t), b(\bar{x}_t)\rangle + \mathbb{E}_\xi \varphi(\bar{x}_t + G(\bar{x}_t, \xi)) - \varphi(\bar{x}_t)\,.$$

Since the Lyapunov function is not the same, we need to explicit here each term. The first term writes:

$$\begin{aligned}
\langle \nabla\varphi(\bar{x}_t), b(\bar{x}_t)\rangle =& \frac{1}{2}\frac{\mathrm{d}A_t}{\mathrm{d}t}\|x_t - x_*\|^2 + \frac{1}{2}\frac{\mathrm{d}B_t}{\mathrm{d}t}\|z_t - x_*\|^2_{H^{-1}} \\
&+ A_t\eta_t\langle x_t - x_*, z_t - x_t\rangle + B_t\eta_t'\langle H^{-1}(z_t - x_*), x_t - z_t\rangle.
\end{aligned}$$

Mimicking the proof of Theorem 2.2, we write

$$\frac{1}{2}\|x_t - x_*\|^2 \le \|x_t - x_*\|^2 - \frac{\mu}{2}\|x_t - x_*\|^2_{H^{-1}},$$

and

$$\begin{aligned}
\langle H^{-1}(z_t - x_*), x_t - z_t\rangle &= \langle z_t - x_*, x_t - x_*\rangle_{H^{-1}} - \|z_t - x_*\|^2_{H^{-1}} \\
&\le \frac{1}{2}\big(\|x_t - x_*\|^2_{H^{-1}} - \|z_t - x_*\|^2_{H^{-1}}\big).
\end{aligned}$$

Hence:

$$\begin{aligned}
\langle \nabla\varphi(\bar{x}_t), b(\bar{x}_t)\rangle \le& \frac{\mathrm{d}A_t}{\mathrm{d}t}\|x_t - x_*\|^2 + \left(B_t\eta_t' - \frac{\mathrm{d}A_t}{\mathrm{d}t}\mu\right)\frac{1}{2}\|x_t - x_*\|^2_{H^{-1}} \\
&+ \left(\frac{\mathrm{d}B_t}{\mathrm{d}t} - B_t\eta_t'\right)\frac{1}{2}\|z_t - x_*\|^2_{H^{-1}} + A_t\eta_t\langle x_t - x_*, z_t - x_t\rangle\,.
\end{aligned}$$

Further,

$$\begin{aligned}
\varphi(\bar{x}_t + G(\bar{x}_t)) - \varphi(\bar{x}_t) =& \frac{A_t}{2}\left(\|x_t - \gamma_t\nabla f(x_t,\xi) - x_*\|^2 - \|x_t - x_*\|^2\right) \\
&+ \frac{B_t}{2}\left(\|(z_t - x_*) - \gamma_t'\nabla f(x_t,\xi)\|^2_{H^{-1}} - \|z_t - x_*\|^2_{H^{-1}}\right)\,.
\end{aligned}$$

Then, expanding and taking expectation over ξ of the first term:

$$\mathbb{E}_\xi\left[\frac{1}{2}\|x_t - \gamma_t\nabla f(x_t,\xi) - x_*\|^2 - \frac{1}{2}\|x_t - x_*\|^2\right]$$

$$= \frac{\gamma_t^2}{2}\mathbb{E}_\xi\left[\|\nabla f(x_t,\xi)\|^2\right] - \gamma_t\langle H(x_t - x_*), x_t - x_*\rangle$$
$$\leq \left(\frac{R^2\gamma_t^2}{2} - \gamma_t\right)\|x_t - x_*\|_H^2,$$

where we used the definition of R^2 in Equation (2.17):

$$\begin{aligned}\mathbb{E}_\xi\left[\|\nabla f(x_t,\xi)\|^2\right] &= (x_t - x_*)^\top\mathbb{E}\left[aa^\top aa^\top\right](x_t - x_*)\\ &= (x_t - x_*)^\top\mathbb{E}\left[\|a\|^2 aa^\top\right](x_t - x_*)\\ &\leq R^2(x_t - x_*)^\top H(x_t - x_*).\end{aligned}$$

The second term writes:

$$\begin{aligned}\frac{1}{2}\mathbb{E}_\xi\left[\|(z_t - x_*) - \gamma_t'\nabla f(x_t,\xi)\|_{H^{-1}}^2 - \|z_t - x_*\|_{H^{-1}}^2\right] &= \frac{\gamma_t'^2}{2}\mathbb{E}_\xi\left[\|\nabla f(x_t,\xi)\|_{H^{-1}}^2\right]\\ &\quad - \gamma_t'\langle x_t - x_*, z_t - x_*\rangle\\ &\leq \frac{\tilde\kappa\gamma_t'^2}{2}\|x_t - x_*\|_H^2\\ &\quad - \gamma_t'\langle x_t - x_*, z_t - x_*\rangle,\end{aligned}$$

where we used the definition of $\tilde\kappa$ in Equation (2.18):

$$\begin{aligned}\mathbb{E}_\xi\left[\|\nabla f(x_t,\xi)\|_{H^{-1}}^2\right] &= (x_t - x_*)^\top\mathbb{E}\left[aa^\top H^{-1}aa^\top\right](x_t - x_*)\\ &= (x_t - x_*)^\top\mathbb{E}\left[a\|a\|_{H^{-1}}^2 a^\top\right](x_t - x_*)\\ &\leq \tilde\kappa(x_t - x_*)^\top H(x_t - x_*).\end{aligned}$$

Combining these inequalities gives the following upper-bound on I_t:

$$\begin{aligned}I_t \leq &\left(\frac{\mathrm{d}A_t}{\mathrm{d}t} - A_t\eta_t\right)\|x_t - x_*\|^2 + \left(\frac{\mathrm{d}B_t}{\mathrm{d}t} - B_t\eta_t'\right)\frac{1}{2}\|z_t - x_*\|_{H^{-1}}^2\\ &+ (A_t\eta_t - B_t\gamma_t')\langle x_t - x_*, z_t - x_*\rangle + \left(B_t\eta_t' - \frac{\mathrm{d}A_t}{\mathrm{d}t}\mu\right)\frac{1}{2}\|x_t - x_*\|_{H^{-1}}^2\\ &+ \left(\tilde\kappa B_t\gamma_t'^2 - A_t\gamma_t\left(2 - R^2\gamma_t\right)\right)\frac{1}{2}\|x_t - x_*\|_H^2\end{aligned}$$

Since $I_t \leq 0$ is still a sufficient condition for ϕ_t to be a supermartingale, we choose parameters such that all prefactors are equal to 0. We first take $\gamma_t = \frac{1}{R^2}$, and we want to satisfy:

$$\frac{\mathrm{d}A_t}{\mathrm{d}t} = A_t\eta_t\,, \qquad \frac{\mathrm{d}B_t}{\mathrm{d}t} = B_t\eta_t' \qquad A_t\eta_t = B_t\gamma_t'\,, \qquad B_t\eta_t' = \frac{\mathrm{d}A_t}{\mathrm{d}t}\mu\,, \qquad B_t\gamma_t'^2 = \frac{A_t}{\tilde\kappa R^2}\,.$$

To satisfy that last equality, we choose:

$$\gamma_t' = \sqrt{\frac{A_t}{B_t\tilde\kappa R^2}}.$$

The rest of the proof then follows just as in the proof of Theorem 2.B.1.

2.C. Proof of Theorem 2.3

By integrating the ODE

$$\begin{aligned} \mathrm{d}x_t &= \eta_t(z_t - x_t)\mathrm{d}t\,, \\ \mathrm{d}z_t &= \eta_t'(x_t - z_t)\mathrm{d}t\,, \end{aligned}$$

between T_k and $T_{k+1}-$, we obtain that there exists τ_k, τ_k'', such that

$$\begin{aligned} \tilde{y}_k = x_{T_{k+1}-} &= x_{T_k} + \tau_k(z_{T_k} - x_{T_k}) = \tilde{x}_k + \tau_k(\tilde{z}_k - \tilde{x}_k)\,, \\ z_{T_{k+1}-} &= z_{T_k} + \tau_k''(x_{T_k} - z_{T_k}) = \tilde{z}_k + \tau_k''(\tilde{x}_k - \tilde{z}_k)\,. \end{aligned} \tag{2.40}$$

From the first equation, we have $\tilde{x}_k = \frac{1}{1-\tau_k}(\tilde{y}_k - \tau_k\tilde{z}_k)$, which gives by substitution in the second equation,

$$\begin{aligned} z_{T_{k+1}-} &= \tilde{z}_k + \tau_k''\left(\frac{1}{1-\tau_k}(\tilde{y}_k - \tau_k\tilde{z}_k) - \tilde{z}_k\right) \\ &= \tilde{z}_k + \tau_k'(\tilde{y}_k - \tilde{z}_k)\,, \end{aligned}$$

where $\tau_k' = \frac{\tau_k''}{1-\tau_k}$.

Further, from (2.4)-(2.5), we obtain the equations

$$\tilde{x}_{k+1} = x_{T_{k+1}} = x_{T_{k+1}-} - \gamma_{T_{k+1}}\nabla f(x_{T_{k+1}-}) = \tilde{y}_k - \gamma_{T_{k+1}}\nabla f(\tilde{y}_k)\,, \tag{2.41}$$

$$\tilde{z}_{k+1} = z_{T_{k+1}} = z_{T_{k+1}-} - \gamma_{T_{k+1}}'\nabla f(x_{T_{k+1}-}) = \tilde{z}_k + \tau_k'(\tilde{y}_k - \tilde{z}_k) - \gamma_{T_{k+1}}'\nabla f(\tilde{y}_k)\,. \tag{2.42}$$

The stated equation (2.10)-(2.12) are the combination of (2.40), (2.41) and (2.42).

1. The parameters of Theorem 2.2.(1) are $\eta_t = \frac{2}{t}, \eta_t' = 0, \gamma_t = \frac{1}{L}$ and $\gamma_t' = \frac{t}{2L}$. In this case, the ODE

$$\begin{aligned} \mathrm{d}x_t &= \eta_t(z_t - x_t)\mathrm{d}t = \frac{2}{t}(z_t - x_t)\mathrm{d}t\,, \\ \mathrm{d}z_t &= \eta_t'(x_t - z_t)\mathrm{d}t = 0\,, \end{aligned}$$

can be integrated in closed form: for $t \geq t_0$,

$$\begin{aligned} x_t &= z_{t_0} + \left(\frac{t_0}{t}\right)^2(x_{t_0} - z_{t_0}) = x_{t_0} + \left(1 - \left(\frac{t_0}{t}\right)^2\right)(z_{t_0} - x_{t_0})\,, \\ z_t &= z_{t_0}\,. \end{aligned}$$

In particular, taking $t_0 = T_k$, $t = T_{k+1}-$, we obtain $\tau_k = 1 - \left(\frac{T_k}{T_{k+1}}\right)^2$, $\tau_k'' = 0$ and thus $\tau_k' = \frac{\tau_k''}{1-\tau_k} = 0$. Finally, $\tilde{\gamma}_k = \gamma_{T_k} = \frac{1}{L}$ and $\tilde{\gamma}_k' = \gamma_{T_k}' = \frac{T_k}{2L}$.

2. The parameters of Theorem 2.2.(2) are $\eta_t = \eta_t' \equiv \sqrt{\frac{\mu}{L}}, \gamma_t \equiv \frac{1}{L}$ and $\gamma_t' \equiv \frac{1}{\sqrt{\mu L}}$. In this case, the ODE

$$\begin{aligned} \mathrm{d}x_t &= \eta_t(z_t - x_t)\mathrm{d}t = \sqrt{\frac{\mu}{L}}(z_t - x_t)\mathrm{d}t\,, \\ \mathrm{d}z_t &= \eta_t'(x_t - z_t)\mathrm{d}t = \sqrt{\frac{\mu}{L}}(x_t - z_t)\mathrm{d}t\,, \end{aligned}$$

can also be integrated in closed form: for $t \geq t_0$,

$$\begin{aligned} x_t &= \frac{x_{t_0} + z_{t_0}}{2} + \frac{x_{t_0} - z_{t_0}}{2} \exp\left(-2\sqrt{\frac{\mu}{L}}(t - t_0)\right) \\ &= x_{t_0} + \frac{1}{2}\left(1 - \exp\left(-2\sqrt{\frac{\mu}{L}}(t - t_0)\right)\right)(z_{t_0} - x_{t_0})\,, \\ z_t &= \frac{x_{t_0} + z_{t_0}}{2} + \frac{z_{t_0} - x_{t_0}}{2} \exp\left(-2\sqrt{\frac{\mu}{L}}(t - t_0)\right) \\ &= z_{t_0} + \frac{1}{2}\left(1 - \exp\left(-2\sqrt{\frac{\mu}{L}}(t - t_0)\right)\right)(x_{t_0} - z_{t_0})\,. \end{aligned}$$

In particular, taking $t_0 = T_k$, $t = T_{k+1}-$, we obtain $\tau_k = \tau_k'' = \frac{1}{2}\left(1 - \exp\left(-2\sqrt{\frac{\mu}{L}}(T_{k+1} - T_k)\right)\right)$ and thus $\tau_k' = \frac{\tau_k''}{1-\tau_k} = \tanh\left(\sqrt{\frac{\mu}{L}}(T_{k+1} - T_k)\right)$. Finally, $\tilde{\gamma}_k = \gamma_{T_k} = \frac{1}{L}$ and $\tilde{\gamma}_k' = \gamma_{T_k}' = \frac{1}{\sqrt{\mu L}}$.

2.D. Continuized Accelerated Coordinate Descent with arbitrary sampling

In this section, we focus on the following problem:

$$\min_{x \in \mathbb{R}^d} f(x), \tag{2.43}$$

where f is of the form $f : x \mapsto g(Rx)$ for some function g and projector $R \in \mathbb{R}^{d \times d}$ (such that $R^2 = R$). We further assume that f is smooth with respect to some matrix $M \in \mathbb{R}^{d \times d}$ and μ-strongly convex with respect to R, *i.e.*:

$$\frac{\mu}{2}\|x - y\|_R^2 \leq f(x) - f(y) - \nabla f(x)^\top (x - y) \leq \frac{1}{2}\|x - y\|_M^2.$$

Note that μ can be equal to zero, but convergence will be slower in this case. We analyze the convergence of the following continuized coordinate descent iteration:

$$\begin{aligned} \mathrm{d}x_t &= \eta_t(z_t - x_t)\mathrm{d}t - \gamma_t \int_\Xi \frac{R_{\xi\xi}}{\mathcal{P}_\xi} \nabla f(x_t, \xi)\mathrm{d}N(t, \xi)\,, \\ \mathrm{d}z_t &= \eta_t'(x_t - z_t)\mathrm{d}t - \gamma_t' \int_\Xi \nabla f(x_t, \xi)\mathrm{d}N(t, \xi)\,, \end{aligned} \tag{2.44}$$

where

$$\nabla f(x_t, \xi) = \frac{1}{\mathcal{P}_\xi} \nabla_\xi f(x_t), \tag{2.45}$$

with the coordinate gradient $\nabla_\xi f(x_t) = e_\xi e_\xi^\top \nabla f(x_t)$, with $e_\xi \in \mathbb{R}^d$ the unit vector associated with coordinate $\xi \in \{1, \ldots, d\}$ and $\mathcal{P}_\xi$ and $\mathrm{d}N$ are defined as in Section 2.7. Note that these iterations are slightly different from the previous stochastic gradient iteration since the stochastic gradient is not the same for x_t and z_t (same direction but different magnitudes). The following theorem is a continuized version of [HBM18], which is itself largely based on [NS17].

Theorem 2.8 (Continuized acceleration of coordinate descent)**.** *Assume that the stochastic gradients are of the coordinate descent form* (2.45)*. Besides, choose parameter L such that:*

$$L \geq \max_{\xi \in \Xi} \frac{M_{\xi\xi} R_{\xi\xi}}{\mathcal{P}_\xi^2}\,. \tag{2.46}$$

Then the continuized acceleration (2.44) *satisfies the following:*

1. *For* $\eta_t = \frac{2}{t}, \eta'_t = 0, \gamma_t = \frac{1}{L}, \gamma'_t = \frac{t}{2L}$,

$$\mathbb{E}f(x_t) - f(x_*) \leq \frac{2L\|z_0 - x_*\|_R^2}{t^2}\,.$$

2. *Assume further that* $\mu > 0$ *and choose the constant parameters* $\eta_t = \eta'_t \equiv \sqrt{\frac{\mu}{L}}$, $\gamma_t \equiv \frac{1}{L}$, $\gamma'_t \equiv \frac{1}{\sqrt{\mu L}}$. *Then ,*

$$\mathbb{E}f(x_t) - f(x_*) \leq \left(f(x_0) - f(x_*) + \frac{\mu}{2}\|z_0 - x_*\|_R^2\right)\exp\left(-\sqrt{\frac{\mu}{L}}t\right)\,.$$

Proof. The proof of this theorem is along the same lines as the proof of Theorem 2.2, and we only highlight the major differences. The process $\bar{x}_t = (t, x_t, z_t)$ satisfies the equation

$$\mathrm{d}\bar{x}_t = b(\bar{x}_t)\mathrm{d}t + \int_\Xi G(\bar{x}_t, \xi)\mathrm{d}N(t,\xi), \quad b(\bar{x}_t) = \begin{pmatrix} 1 \\ \eta_t(z_t - x_t) \\ \eta'_t(x_t - z_t)\end{pmatrix}, \quad G(\bar{x}_t, \xi) = \begin{pmatrix} 0 \\ -\gamma_t \frac{R_{\xi\xi}}{\mathcal{P}_\xi}\nabla f(x_t, \xi) \\ -\gamma'_t \nabla f(x_t, \xi)\end{pmatrix}.$$

We also consider a slightly different Lyapunov function ϕ_t that takes into account the projector R:

$$\phi_t = A_t\left(f(x_t) - f_*\right) + \frac{B_t}{2}\|z_t - x_*\|_R^2$$

This change of norm is essential to take into account the fact that f is not strongly convex with respect to the euclidean norm, but only with respect to $\|\cdot\|_R$. We apply Proposition 2.A.2 to $\phi_t = \varphi(\bar{x}_t) = \varphi(t, x_t, z_t)$ and obtain

$$\phi_t = \phi_0 + \int_0^t I_s \mathrm{d}s + M_t\,, \tag{2.47}$$

where M_t is a martingale and

$$I_t = \langle \nabla\varphi(\bar{x}_t), b(\bar{x}_t)\rangle + \mathbb{E}_\xi\varphi(\bar{x}_t + G(\bar{x}_t, \xi)) - \varphi(\bar{x}_t)\,.$$

The computation of the first term remains the same: the inequality (2.32)-(2.33) holds. The computation of the second term becomes

$$\begin{aligned}\mathbb{E}_\xi\varphi(\bar{x}_t + G(\bar{x}_t,\xi)) - \varphi(\bar{x}_t) = A_t\left(\mathbb{E}_\xi f\left(x_t - \gamma_t\frac{R_{\xi\xi}}{\mathcal{P}_\xi}\nabla f(x_t,\xi)\right) - f(x_t)\right) \\ + \frac{B_t}{2}\left(\mathbb{E}_\xi\|(z_t - x_*) - \gamma'_t\nabla f(x_t,\xi)\|_R^2 - \|z_t - x_*\|_R^2\right)\,.\end{aligned}$$

As f is M-smooth,

$$f\left(x_t - \gamma_t\frac{R_{\xi\xi}}{\mathcal{P}_\xi}\nabla f(x_t,\xi)\right) - f(x_t) \leq \langle\nabla f(x_t), -\gamma_t\frac{R_{\xi\xi}}{\mathcal{P}_\xi}\nabla f(x_t,\xi)\rangle + \frac{1}{2}\|\gamma_t\frac{R_{\xi\xi}}{\mathcal{P}_\xi}\nabla f(x_t,\xi)\|_M^2\,.$$

In the additive case, the variance is bounded by σ^2. In this case, we have that:

$$\|\frac{R_{\xi\xi}}{\mathcal{P}_\xi}\nabla f(x_t,\xi)\|_M^2 = \frac{M_{\xi\xi}R_{\xi\xi}}{\mathcal{P}_\xi^2}\|\nabla f(x_t,\xi)\|_R^2 \leq L\|\nabla f(x_t,\xi)\|_R^2, \tag{2.48}$$

and similarly:

$$\langle \nabla f(x_t), -\gamma_t \frac{R_{\xi\xi}}{\mathcal{P}_\xi} \nabla f(x_t, \xi) \rangle = -\gamma_t \frac{R_{\xi\xi}}{\mathcal{P}_\xi^2} \|\nabla_\xi f(x_t)\|^2 = \gamma_t \|\nabla f(x_t, \xi)\|_R^2. \tag{2.49}$$

Thus:

$$\mathbb{E}_\xi f\left(x_t - \gamma_t \frac{R_{\xi\xi}}{\mathcal{P}_\xi} \nabla f(x_t, \xi)\right) - f(x_t) \leq \gamma_t(1 - \gamma_t L)\mathbb{E}_\xi \|\nabla f(x_t, \xi)\|_R^2 .$$

Similarly, thanks to the unbiasedness of $\nabla f(x_t, \xi)$,

$$\begin{aligned}\mathbb{E}_\xi \|(z_t - x_*) - \gamma_t' \nabla f(x_t, \xi)\|_R^2 - \|z_t - x_*\|_R^2 \\ &= -2\gamma_t' \langle \mathbb{E}_\xi R \nabla f(x_t, \xi), z_t - x_* \rangle + \gamma_t'^2 \mathbb{E}_\xi \|\nabla f(x_t, \xi)\|_R^2 \\ &\leq -2\gamma_t' \langle \nabla f(x_t), z_t - x_* \rangle + \gamma_t'^2 \mathbb{E}_\xi \|\nabla f(x_t, \xi)\|_R^2 .\end{aligned}$$

This gives

$$\begin{aligned}\varphi(\bar{x}_t + G(\bar{x}_t)) - \varphi(\bar{x}_t) \leq -B_t \gamma_t' \langle \nabla f(x_t), z_t - x_* \rangle \\ + \left(B_t \gamma_t'^2 - A_t \gamma_t \left(2 - L\gamma_t\right)\right) \frac{1}{2} \mathbb{E}_\xi \|\nabla f(x_t, \xi)\|_R^2 .\end{aligned}$$

Combining the bounds, we obtain

$$\begin{aligned}I_t \leq \left(\frac{\mathrm{d}A_t}{\mathrm{d}t} - A_t \eta_t\right) \langle \nabla f(x_t), x_t - x_* \rangle + \left(\frac{\mathrm{d}B_t}{\mathrm{d}t} - B_t \eta_t'\right) \frac{1}{2} \|z_t - x_*\|_R^2 \\ + (A_t \eta_t - B_t \gamma_t') \langle \nabla f(x_t), z_t - x_* \rangle + \left(B_t \eta_t' - \frac{\mathrm{d}A_t}{\mathrm{d}t} \mu\right) \frac{1}{2} \|x_t - x_*\|_R^2 \\ + \left(B_t \gamma_t'^2 - A_t \gamma_t \left(2 - L\gamma_t\right)\right) \frac{1}{2} \mathbb{E}_\xi \|\nabla f(x_t, \xi)\|_R^2 .\end{aligned}$$

We see that we obtain a result that is very similar to that of the deterministic case. The choices of parameters of Theorem 2.8 cancel all first five prefactors, and satisfy $\gamma_t = \frac{1}{L}$, $A_t L \gamma_t^2 = B_t \gamma_t'^2$. We thus obtain $I_t \leq 0$ and so ϕ_t is a supermartingale, and the rest follows as in Appendix 2.B.1. □

2.E. Proof of Theorem 2.6

Proof. First note that the Hessian of the dual objective writes for some $\lambda \in \mathbb{R}^{|\mathcal{E}| \times d}$:

$$A^\top \nabla^2 F^*(A\lambda) A \succcurlyeq \frac{1}{L} A^\top A, \tag{2.50}$$

since F^* is L^{-1} strongly-convex when F is L-smooth [KSST09]. Thus, the dual objective is μ_{gossip}/L strongly convex on the orthogonal of the kernel of A. Similarly, the smoothness of the dual objective in direction $\{v, w\}$ is equal to:

$$M_{\{v,w\}\{v,w\}} = e_{\{v,w\}}^\top A^\top \nabla^2 F^*(A\lambda) A e_{\{v,w\}} \preccurlyeq \frac{1}{\mu} e_{\{v,w\}}^\top A^\top A e_{\{v,w\}} = \frac{\mathcal{P}_{\{v,w\}}}{2\mu}. \tag{2.51}$$

Thus, we have that:

$$L_{\text{dual}} = \max_{\{v,w\}} \frac{M_{\{v,w\}\{v,w\}} R\{v,w\}}{\mathcal{P}_{\{v,w\}}^2} = \frac{1}{\mu} \max_{\{v,w\}} \frac{R\{v,w\}}{\mathcal{P}_{\{v,w\}}}. \tag{2.52}$$

Then, the result follows directly from applying Theorem (2.8), together with the smoothness of the dual gradients, since:

$$\mathbb{E}\sum_{v\in V}\frac{1}{2}\|\nabla f_v^*(z_t(v)) - x_\star\|^2 \leq \mathbb{E}\frac{1}{2\mu}\|A\lambda_t^{(z)} - A\lambda_\star\|^2 \leq \frac{\lambda_{\max}(AA^\top)}{2\mu}\mathbb{E}\|\lambda_t^{(z)} - \lambda_\star\|_R^2\,. \tag{2.53}$$

□

Chapter 3

The asynchronous speedup of Asynchronous SGD

The previous chapter introduced the continuized framework: it allows decentralized agents to share a continous-time clock, bypassing the limits of a shared iteration counters that requires synchrony between users. However, the continuized acceleration we provided does not cover delays, a core component of asynchrony. In the following three chapters, we investigate the effect of communication and computation delays over optimization rates, and in particular ask the question: *when is asynchrony beneficial in terms of compute speed ?* We start in the simplest case: the centralized setting, with homogeneous workers first and then heterogeneous ones, to first show that Asynchronous SGD is always faster than its synchronous counterpart (Minibatch SGD), and to provide an explicit *asynchronous speedup*, that quantifies how faster asynchrony is in this context. Quantitatively, in order to reach a small given precision ε, for M machines, where machine m needs s_m seconds to compute gradients and communicate with the server, we show that Asynchronous SGD is

$$\frac{1}{M}\sum_{m=1}^{M}\frac{s_{\max}}{s_m}$$

times faster than Minibatch SGD.

The existing analysis of asynchronous stochastic gradient descent (SGD) degrades dramatically when any delay is large, giving the impression that performance depends primarily on the delay. On the contrary, we prove much better guarantees for the same asynchronous SGD algorithm regardless of the delays in the gradients, depending instead just on the number of parallel devices used to implement the algorithm. Our guarantees are strictly better than the existing analyses, and we also argue that asynchronous SGD outperforms synchronous minibatch SGD in the settings we consider. For our analysis, we introduce a novel recursion based on "virtual iterates" and delay-adaptive stepsizes, which allow us to derive state-of-the-art guarantees for both convex and non-convex objectives.

3.1. Introduction

In this chapter, we primarily focus on *homogeneous* workers: they all have access to stochastic gradients of the same function F. We therefore use the notation m for workers and denote as M the number of these workers or *machines*, to emphasize that the setting slightly differs from other chapters. We consider solving stochastic optimization problems of the form

$$\min_{\mathbf{x}\in\mathbb{R}^d}\ \{F(\mathbf{x}) = \mathbb{E}_{\xi\sim\mathcal{D}} f(\mathbf{x};\xi)\}, \tag{3.1}$$

which includes machine learning (ML) training objectives, where $f(\mathbf{x};\xi)$ represents the loss of a model parameterized by $\mathbf{x}$ on the datum ξ. Depending on the application, $\mathcal{D}$ could

represent a finite dataset of size n or a population distribution. In recent years, such stochastic optimization problems have continued to grow rapidly in size, both in terms of the dimension d of the optimization variable—i.e., the number of model parameters in ML—and in terms of the quantity of data—i.e., the number of samples $\xi_1, \dots, \xi_n \sim \mathcal{D}$ being used. With d and n regularly reaching the tens or hundreds of billions, it is increasingly necessary to use parallel optimization algorithms to handle the large scale and to benefit from data stored on different machines.

There are many ways of employing parallelism to solve Equation (3.1), but the most popular approaches in practice are first-order methods based on stochastic gradient descent (SGD). At each iteration, SGD employs stochastic estimates of ∇F to update the parameters as $\mathbf{x}_k = \mathbf{x}_{k-1} - \gamma_k \nabla f(\mathbf{x}_{k-1}; \xi_{k-1})$ for an i.i.d. sample $\xi_{k-1} \sim \mathcal{D}$. Given M machines capable of computing these stochastic gradient estimates $\nabla f(\mathbf{x}; \xi)$ in parallel, one approach to parallelizing SGD is "Distributed SGD", or "Minibatch SGD" (Algorithm 1.1). This refers to a synchronous, parallel algorithm that dispatches the current parameters $\mathbf{x}_{k-1}$ to each of the M machines, waits while they compute and communicate back their gradient estimates $\mathbf{g}_{k-1}^1, \dots, \mathbf{g}_{k-1}^M$, and then takes a Minibatch SGD step $\mathbf{x}_k = \mathbf{x}_{k-1} - \gamma_k \cdot \frac{1}{M} \sum_{m=1}^M \mathbf{g}_{k-1}^m$, as explained in detailed in Section 1.2.1.

However, since Minibatch SGD waits for all M of the machines to finish computing their gradient estimates before updating, it proceeds only at the speed of the *slowest* machine. There are several possible sources of delays: nodes may have heterogeneous hardware with different computational throughputs [KMA+19, HLA+21], network latency can slow the communication of gradients, and nodes may even just drop out [RGPP21]. Slower "straggler" nodes can arise in many natural parallel settings including training ML models using multiple GPUs [CPM+16] or in the cloud, and sensitivity to these stragglers poses a serious problem for Minibatch SGD and other similar synchronous algorithms.

3.1.1. Asynchronous SGD

In this work, we consider a different, *asynchronous* parallel variant of SGD, which we define in Algorithm 3.1 and which has a long history [NBB01, AD11, ASS20]. For this method, whenever one of the M machines finishes computing a stochastic gradient, the algorithm immediately uses it to take an SGD step, and then that machine begins computing a new stochastic gradient at the newly updated parameters. Because of the asynchronous updates, the other machines are now estimating the gradient at *out-of-date* parameters, so this algorithm ends up performing updates of the form[1]

$$\mathbf{x}_k = \mathbf{x}_{k-1} - \gamma_k \nabla f(\mathbf{x}_{k-\tau(k)}; \xi_{k-\tau(k)}),$$

where $\tau(k)$ is the "delay" of the gradient at iteration k, which is often much greater than one. Nevertheless, even though the updates are not necessarily well-aligned with the gradient of F at the current parameters, the delays are usually not a huge problem in practice [DCM+12]. Asynchronous SGD has been particularly popular in reinforcement learning applications [MBM+16, NSB+15] and federated learning [CCA+21, NMZ+22], providing significant speed-ups over Minibatch SGD.

However and as explained in Section 1.3, the existing theoretical guarantees for Asynchronous SGD are disappointing, and the typical approach to analyzing the algorithm involves assuming that all of the delays are either the same, $\tau(k) = \tau$, or at least upper bounded, $\tau(k) \le \tau_{\max}$ [AD11, MPP+17, LPLJ18, ASS20, SK20]. These analyses then show that the number of updates needed to reach accuracy ϵ grows linearly with $\tau_{\max}$, which could be very

[1] Although this algorithm is asynchronous in the sense that different workers will have un-synchronized iterates, we nevertheless focus on a situation where each SGD step is an atomic/locked update of the parameters $\mathbf{x}_k$. This is in contrast to methods using lock-free updates, e.g., in the style of *Hogwild!* [RRWN11], where different coordinates of the parameters might be updated and overwritten simultaneously by different workers.

Algorithm 3.1: Asynchronous SGD

Input: initialization $\mathbf{x}_0 \in \mathbb{R}^d$, stepsizes $\gamma_k > 0$
Each worker $m \in [M]$ begins calculating $\nabla f(\mathbf{x}_0; \xi_0^m)$
for $k = 1, 2, \ldots$ **do**
  Gradient $\nabla f(\mathbf{x}_{\text{prev}(k,m_k)}; \xi_{\text{prev}(k,m_k)}^{m_k})$ arrives from some worker m_k
  Update: $\mathbf{x}_k = \mathbf{x}_{k-1} - \gamma_k \nabla f(\mathbf{x}_{\text{prev}(k,m_k)}; \xi_{\text{prev}(k,m_k)}^{m_k})$
  Send $\mathbf{x}_k$ to worker m_k, which begins calculating $\nabla f(\mathbf{x}_k; \xi_k^{m_k})$
end for

painful for heterogeneous workers, as the toy example in Section 1.3 shows.

3.1.2. The asynchronous speedup of Asynchronous SGD and speedup over Minibatch SGD

We will frequently compare Asynchronous SGD to Minibatch SGD (*e.g.*, Table 3.2), and to make this comparison easier, suppose for simplicity that each worker requires a fixed time of s_m seconds per gradient computation, so in S seconds, each machine computes $\frac{S}{s_m}$ stochastic gradients. Importantly, this translates into drastically different numbers of parameter updates for Asynchronous versus Minibatch SGD: the former takes one step per gradient computed, while the latter only takes one step for each gradient from the *slowest* machine. That is, Asynchronous and Minibatch SGD take

$$K_{\text{Async}} = \sum_{m=1}^{M} \frac{S}{s_m} \qquad \text{and} \qquad K_{\text{Mini}} = \min_{1 \le m \le M} \frac{S}{s_m} \tag{3.2}$$

total steps, respectively. So, it is easy to see that Asynchronous SGD takes *at least* M times more steps than Minibatch SGD in any fixed amount of time, and even more than that when the machines have varying speeds. Soon, we will prove guarantees for Asynchronous SGD that match the guarantee for Minibatch SGD using exactly M times fewer updates, meaning that our Asynchronous SGD guarantees are strictly better than the Minibatch SGD guarantees in terms of runtime. Concretely, given a budget time T, Asynchronous SGD is

$$\frac{1}{M} \sum_{m=1}^{M} \frac{s_{\max}}{s_m}$$

times faster than Minibatch SGD: this quantitative speedup is what we call the asynchronous speedup of Asynchronous SGD.

Structure of this chapter. In this chapter, we provide a new analysis for Asynchronous SGD, described in Section 3.2, which we use to prove better convergence guarantees. In contrast to the existing guarantees that are based on $\tau_{\max}$, ours depend only on the number of workers, M, and show that Asynchronous SGD is better than the Minibatch SGD algorithm described earlier. For Lipschitz-continuous objectives, our results in Section 3.3 improve over existing Asynchronous and Minibatch SGD guarantees, and in the non-smooth, convex setting they are, in fact, minimax optimal. In Section 3.4, we prove state-of-the-art guarantees for smooth losses, which are summarized in Table 3.1. We do this by introducing a novel delay-adaptive stepsize schedule $\gamma_k \sim 1/\tau(k)$. The high-level intuition behind our proofs is that, although *some* of the gradients may have very large delay, *most* of the gradients have delay $\mathcal{O}(M)$, which is enough for good performance. Finally, in Section 3.5 we generalize our analysis to heterogenous objective functions.

Table 3.1 – *Comparison of the convergence rates for smooth objectives in terms of K, the total number of stochastic gradients used. For Minibatch SGD with R updates with minibatch size M, it holds $K = MR$. For simplicity, we ignore all logarithmic and constant terms, including the coefficients in front of the exponents. The stated rates are upper bounds on $\mathbb{E}\left[\|\nabla F(\mathbf{x})\|^2\right]$ in the non-convex case, and $\mathbb{E}\left[F(\mathbf{x}) - F^*\right]$ in the (strongly) convex case.*

Method and reference	Convex	Strongly Convex	Non-Convex
Minibatch SGD[(a)] [GLQ+19] [KR20]	$\frac{M}{K} + \frac{\sigma}{\sqrt{K}}$	$e^{\frac{-\mu K}{LM}} + \frac{\sigma^2}{K}$	$\frac{M}{K} + \frac{\sigma}{\sqrt{K}}$
Asynchronous SGD (fixed delay τ) [SK20]	$\frac{\tau}{K} + \frac{\sigma}{\sqrt{K}}$	$e^{\frac{-\mu K}{L\tau}} + \frac{\sigma^2}{K}$	$\frac{\tau}{K} + \frac{\sigma}{\sqrt{K}}$
Asynchronous SGD (arbitrary delays) **Our work**	$\frac{M}{K} + \frac{\sigma}{\sqrt{K}}$	$e^{\frac{-\mu K}{LM}} + \frac{\sigma^2}{K}$	$\frac{M}{K} + \frac{\sigma}{\sqrt{K}}$

[(a)] [GLQ+19] analyzed SGD in the strongly convex regime and [KR20] in the non-convex regime.

Related works specific to this chapter. Closely related to this chapter, [MPP+17] proposed and utilized the analysis tool of "virtual iterates" for Asynchronous SGD under bounded delays. [SK20] extended these results, albeit restricting delays to be constant, and [LPLJ18] considered lock-free updates. We use the same proof approach, but with a different virtual sequence and different, delay-adaptive stepsizes.

In a concurrent work, [KSJ22] used a similar technique to ours to study Asynchronous SGD with potentially unbounded delays. Their bounds are stated using empirical average of the delays rather than M. Compared to the work of [KSJ22], our theory includes guarantees for non-smooth problems as given in Theorem 3.1. They, on the other hand, have an extra result for the case of constant stepsize and bounded delays without assuming bounded gradients. Our Theorem 3.2 that covers non-convex, convex, and strongly convex problems has almost the same delay-adaptive stepsize as their Theorem 8 that covers non-convex functions only. For heterogeneous data, we study standard Asynchronous SGD, whereas [KSJ22] used a special scheduling procedure to balance the workers, see also the comments after our Theorem 3.3.

3.1.3. Notation and problem setting

We consider solving the problem (3.1) under several standard [Bub15, see, e.g.,] combinations of conditions on the objective F. We denote the minimum of F as $F^* = \min_{\mathbf{x}} F(\mathbf{x})$, an upper bound on the **initial suboptimality** as $\Delta \geq F(\mathbf{x}_0) - F^*$, and an upper bound on the **initial distance** to the minimizer as $B \geq \min\{\|\mathbf{x}_0 - \mathbf{x}^*\| : \mathbf{x}^* \in \arg\min_{\mathbf{x}} F(\mathbf{x})\}$. Additionally to the definitions of Section 1.6, we may also assume the stochastic gradients have σ^2-**bounded variance**, meaning that for all $\mathbf{x}$, $\mathbb{E}_{\xi\sim\mathcal{D}}\|\nabla f(\mathbf{x};\xi) - \nabla F(\mathbf{x})\|^2 \leq \sigma^2$.

Finally, we mainly focus on "homogeneous" optimization, where each machine computes each stochastic gradient using an i.i.d. sample $\xi \sim \mathcal{D}$. This is in contrast to the "heterogeneous" setting, where different machines have access to data drawn from different sources, meaning that stochastic gradients estimated on different machines can have different distributions (see Section 3.5).

Delay notation. The gradients used by Algorithm 3.1 may arrive out of order, so the parameters $\mathbf{x}_k$ at iteration k will often be updated using stochastic gradients $\nabla f(\mathbf{x}_j;\xi_j)$ evaluated at out-of-date parameters $\mathbf{x}_j$ for $j < k-1$; we therefore introduce additional notation for describing the delays. We use $m_k \in [M]$ to denote the index of the worker whose stochastic gradient estimate is used in iteration k to compute $\mathbf{x}_k$. In addition, for each iteration k and worker m, we introduce

$$\mathrm{prev}(k,m) = \max\{j < k : m_j = m\} \qquad \text{and} \qquad \mathrm{next}(k,m) = \min\{j \geq k : m_j = m\},$$

which denotes the index of the last iteration before k when machine m returned a gradient, and the index of the first iteration after k (inclusive) when machine m will return a gradient, respectively. Accordingly, at iteration k, machine m is in the process of estimating $\nabla f(\mathbf{x}_{\mathrm{prev}(k,m)};\xi^m_{\mathrm{prev}(k,m)})$. We define the current "delay" of this gradient as the number of iterations that have happened since $\mathrm{prev}(k,m)$, i.e., $\tau(k,m) = k - \mathrm{prev}(k,m)$. Abusing notation, we shorten to $\tau(k) = \tau(k,m_k)$ for the delay of the gradient used to compute $\mathbf{x}_k$.

3.2. Analysis of Asynchronous SGD via virtual iterates

The central idea in our analysis is to focus on a virtual iterate sequence, which tracks, roughly, how the parameters would have evolved if there were no delays. We note that this sequence is only used for the purpose of analysis, and is never actually computed. This technique is related to previous approaches [MPP+17, LPLJ18, SK20], with the key difference being *which* virtual sequence we track. Specifically, in addition to $\mathbf{x}_0,\dots,\mathbf{x}_K$—the actual sequence of iterates generated by Algorithm 3.1—we introduce the complementary sequence $\hat{\mathbf{x}}_1,\dots,\hat{\mathbf{x}}_K$ which evolves according to

$$\begin{gathered}\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k - \hat{\gamma}_k \nabla f(\mathbf{x}_k;\xi_k^{m_k}),\\ \text{where } \hat{\mathbf{x}}_1 = \mathbf{x}_0 - \sum_{m=1}^{M} \gamma_{\mathrm{next}(1,m)} \nabla f(\mathbf{x}_0;\xi_0^m) \quad \text{and} \quad \hat{\gamma}_k = \gamma_{\mathrm{next}(k+1,m_k)}\,.\end{gathered} \tag{3.3}$$

This virtual sequence $\hat{\mathbf{x}}_{k+1}$ evolves *almost* according to SGD (without delays), although we note that it uses gradients evaluated at $\mathbf{x}_k$ rather than $\hat{\mathbf{x}}_k$. The stepsize used for this update, $\hat{\gamma}_k$, is the stepsize that is eventually used by Algorithm 3.1 when it takes a step using the gradient evaluated at $\mathbf{x}_k$. The core of our proofs is showing that $\mathbf{x}_k$ and $\hat{\mathbf{x}}_k$ remain close using the following Lemma:

Lemma 3.2.1. *Let $\{\mathbf{x}_k\}$ and $\{\hat{\mathbf{x}}_k\}$ be defined as in Algorithm 3.1 and (3.3), respectively. Then for all $k \geq 1$*

$$\mathbf{x}_k - \hat{\mathbf{x}}_k = \sum_{m\in[M]\setminus\{m_k\}} \gamma_{\mathrm{next}(k,m)} \nabla f(\mathbf{x}_{\mathrm{prev}(k,m)};\xi^m_{\mathrm{prev}(k,m)})\,.$$

Proof. First, we expand the update of $\mathbf{x}_k$ in Algorithm 3.1, and of $\hat{\mathbf{x}}_k$ in (3.3). Denoting $\mathbf{e}_k = \mathbf{x}_k - \hat{\mathbf{x}}_k$,

$$\begin{aligned}\mathbf{e}_k &= \mathbf{e}_{k-1} - \gamma_k \nabla f(\mathbf{x}_{\mathrm{prev}(k,m_k)};\xi^{m_k}_{\mathrm{prev}(k,m_k)}) + \hat{\gamma}_{k-1}\nabla f(\mathbf{x}_{k-1};\xi_{k-1}^{m_{k-1}})\\ &= \mathbf{e}_1 - \sum_{j=2}^{k} \gamma_j \nabla f(\mathbf{x}_{\mathrm{prev}(j,m_j)};\xi^{m_j}_{\mathrm{prev}(j,m_j)}) + \sum_{j=1}^{k-1} \hat{\gamma}_j \nabla f(\mathbf{x}_j;\xi_j^{m_j})\\ &= \sum_{m=1}^{M} \gamma_{\mathrm{next}(1,m)} \nabla f(\mathbf{x}_0;\xi_0^m) - \sum_{j=1}^{k} \gamma_j \nabla f(\mathbf{x}_{\mathrm{prev}(j,m_j)};\xi^{m_j}_{\mathrm{prev}(j,m_j)}) + \sum_{j=1}^{k-1} \hat{\gamma}_j \nabla f(\mathbf{x}_j;\xi_j^{m_j})\,.\end{aligned}$$

From here, we note that the second term, which comprises all of the gradients used by

Algorithm 3.1 to make the first k updates, can be rewritten as:

$$-\sum_{m=1}^{M}\gamma_{\text{next}(1,m)}\nabla f(\mathbf{x}_0;\xi_0^m)\mathbf{1}_{\{\text{next}(1,m)\le k\}}-\sum_{i=1}^{k-1}\hat{\gamma}_i\nabla f(\mathbf{x}_i;\xi_i^{m_i})\mathbf{1}_{\{\text{next}(i,m_i)\le k\}}\,,$$

and substituting into the expression for $\mathbf{e}_k$ above, there are k cancellations and the claim follows. □

How does this help us? For all of our results, our strategy is to show that the virtual iterates $\hat{\mathbf{x}}_k$ evolve essentially according to SGD (without delays), that $\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|$ remains small throughout the algorithm's execution, and therefore that the Asynchronous SGD iterates $\mathbf{x}_k$ are nearly as good as SGD without delays. Lemma 3.2.1 is key for the second step. Whereas previous work tries to bound $\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|$ by reasoning about the delays involved in the first k updates, we observe that $\mathbf{x}_k-\hat{\mathbf{x}}_k$ is just the sum of $M-1$ gradients, so our bound naturally incurs a dependence on M, but it is not directly affected by the delays themselves.

3.3. Convergence guarantees for Lipschitz losses

We begin by analyzing Algorithm 3.1 for convex, Lipschitz-continuous losses.

Theorem 3.1. *Let the objective F be convex, let $f(\cdot;\xi)$ be G-Lipschitz-continuous for each ξ, and let there be a minimizer $\mathbf{x}^*\in\arg\min_{\mathbf{x}}F(\mathbf{x})$ for which $\|\mathbf{x}_0-\mathbf{x}^*\|\le B$. Then for any number of iterations[2] $K\ge M$, Algorithm 3.1 with constant stepsize $\gamma_k=\gamma=B/(G\sqrt{KM})$ ensures*

$$\mathbb{E}\left[F\left(\frac{1}{K}\sum_{k=1}^{K}\mathbf{x}_k\right)-F^*\right]\le\frac{3GB\sqrt{M}}{\sqrt{K}}\,.$$

Proof. Let $\mathbf{x}^*\in\arg\min_{\mathbf{x}}F(\mathbf{x})$ with $\|\mathbf{x}_0-\mathbf{x}^*\|\le B$. First, we follow the typical analysis of stochastic gradient descent [Bub15, see, Theorem 3.2, e.g.] by expanding the update of $\hat{\mathbf{x}}_{k+1}$ from (3.3):

$$\begin{aligned}\mathbb{E}\|\hat{\mathbf{x}}_{k+1}-\mathbf{x}^*\|^2&=\mathbb{E}\Big[\|\hat{\mathbf{x}}_k-\mathbf{x}^*\|^2+\gamma^2\big\|\nabla f(\mathbf{x}_k;\xi_k^{m_k})\big\|^2-2\gamma\langle\nabla F(\mathbf{x}_k),\hat{\mathbf{x}}_k-\mathbf{x}^*\rangle\Big]\\&\le\mathbb{E}\Big[\|\hat{\mathbf{x}}_k-\mathbf{x}^*\|^2+\gamma^2G^2-2\gamma[F(\mathbf{x}_k)-F^*]+2\gamma\langle\nabla F(\mathbf{x}_k),\mathbf{x}_k-\hat{\mathbf{x}}_k\rangle\Big]\\&\le\mathbb{E}\Big[\|\hat{\mathbf{x}}_k-\mathbf{x}^*\|^2+\gamma^2G^2-2\gamma[F(\mathbf{x}_k)-F^*]+2\gamma G\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|\Big].\end{aligned}$$

For the first inequality, we used the convexity of F and that $f(\cdot;\xi)$ being G-Lipschitz implies $\|\nabla f(\mathbf{x};\xi)\|\le G$ for all $\mathbf{x}$; for the second, we again used the G-Lipschitzness of $f(\cdot;\xi)$ along with the Cauchy-Schwarz inequality. Continuing as in the standard SGD analysis, we rearrange the expression, average over K, apply the convexity of F, and telescope the sum to conclude:

$$\mathbb{E}\left[F\left(\frac{1}{K}\sum_{k=1}^{K}\mathbf{x}_k\right)-F^*\right]\le\mathbb{E}\left[\frac{\|\hat{\mathbf{x}}_1-\mathbf{x}^*\|^2}{2\gamma K}+\frac{\gamma G^2}{2}+\frac{G}{K}\sum_{k=1}^{K}\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|\right].\tag{3.4}$$

The first two terms almost exactly match the guarantee of SGD with fixed stepsize γ. The main difference—and the place where Lemma 3.2.1 and the Lipschitzness of the losses plays a key role—is in bounding the third term. Since $\left\|\nabla f(\mathbf{x}_{\text{prev}(k,m)};\xi^m_{\text{prev}(k,m)})\right\|\le G$, we just

[2] W.l.o.g. we can take $M\le K$ because at most K of the workers are actually able participate in the first K updates of Algorithm 3.1, and machines that do not participate can simply be ignored in the analysis.

Table 3.2 – *We compare optimization terms in the smooth and convex setting and the resulting speed-ups in the fixed-computation-speed model of Section 3.1.2. We denote* $s_{\max} = \max_m s_m$ *and approximate the maximum delay as* $\tau_{\max} = \sum_{m=1}^{M} s_{\max}/s_m$*. The speedup is the largest factor* α *such that Asynchronous SGD attains the same error in* S *seconds as Minibatch SGD would in* αS *seconds.*

Method	# of Updates	Optimization Term	Speedup
Minibatch SGD	$R = \frac{S}{s_{\max}}$	$\mathcal{O}\left(\frac{1}{R}\right)$	1
Asynchronous SGD (prior works)	$K = \sum_{m=1}^{M} \frac{S}{s_m}$	$\mathcal{O}\left(\frac{\tau_{\max}}{K}\right)$	1
Asynchronous SGD (our work)	$K = \sum_{m=1}^{M} \frac{S}{s_m}$	$\mathcal{O}\left(\frac{M}{K}\right)$	$\frac{1}{M}\sum_{m=1}^{M} \frac{s_{\max}}{s_m} \geq 1$

use the triangle inequality:

$$\|\mathbf{x}_k - \hat{\mathbf{x}}_k\| = \left\| \sum_{m \in [M] \setminus \{m_k\}} \gamma \nabla f(\mathbf{x}_{\text{prev}(k,m)}; \xi^m_{\text{prev}(k,m)}) \right\| \leq (M-1)\gamma G. \tag{3.5}$$

Combining this with $\|\hat{\mathbf{x}}_1 - \mathbf{x}^*\|^2 = \left\|\mathbf{x}_0 - \gamma \sum_{m=1}^{M} \nabla f(\mathbf{x}_0; \xi_0^m) - \mathbf{x}^*\right\|^2 \leq 2B^2 + 2\gamma^2 M^2 G^2$, and plugging in our stepsize in (3.4) completes the proof. □

To understand this result, it is instructive to recall the worst-case performance of K_{Mini} steps of Minibatch SGD [NY83] in the setting of Theorem 3.1:

$$\mathbb{E}[F(\mathbf{x}_{\text{Mini}}) - F^*] = \mathcal{O}\left(GB/\sqrt{K_{\text{Mini}}}\right). \tag{3.6}$$

From this, we see that our guarantee for Asynchronous SGD in Theorem 3.1 matches the rate for $K_{\text{Mini}} = K/M$ steps of Minibatch SGD. Furthermore, at least in the simplified model of Section 3.1.2, Asynchronous SGD takes at least M times more steps than Minibatch SGD in a given span of time, and therefore Theorem 3.1 guarantees better performance than (3.6) in terms of runtime. Moreover, previous analyses of Asynchronous SGD [SK20, e.g.,] provide guarantees with $\tau_{\max}$ replacing M in our bound. Since necessarily $\tau_{\max} \geq M$, this means that our guarantee is never worse than the existing ones and it can be much better, for example, in a case where one severe straggler results in $\tau_{\max} \approx K$ but $M \ll K$. In fact, our guarantee in Theorem 3.1 is minimax optimal in the setting considered [WWS+18, Section 4.3].

Finally, we emphasize that although M appears in the numerator of the error guarantee in Theorem 3.1, this does *not* mean that the guarantee necessarily degrades when more parallel workers are added. In particular, adding more workers always means that more gradients will be calculated in any given amount of runtime. More concretely, in the model of Section 3.1.2 where the machines have fixed speeds, the expression for $K_{\text{Async}} = K_{\text{Async}}(S, s_1, \ldots, s_m)$ in (3.2) implies that adding an $(M+1)^{\text{th}}$ machine gives a better guarantee whenever s_{M+1} is smaller than the harmonic mean of $s_1, \ldots, s_M$:

$$s_{M+1} \leq \left(\frac{1}{M}\sum_{m=1}^{M} \frac{1}{s_m}\right)^{-1} \implies \frac{M+1}{K_{\text{Async}}(S, s_1, \ldots, s_{M+1})} \leq \frac{M}{K_{\text{Async}}(S, s_1, \ldots, s_M)}.$$

Remark 3.3.1. *Following the same high-level approach, we can also analyze Algorithm 3.1 with constant stepsizes for non-convex objectives but smooth-Lipschitz losses: such a result is present in [MBEW22, Theorem 2].*

3.4. Convergence guarantees for non-Lipschitz losses

Now, we analyze smooth but not necessarily Lipschitz-continuous losses. The previous proofs relied crucially on the gradients being bounded in norm in order to control $\mathbf{x}_k - \hat{\mathbf{x}}_k$. For general smooth losses, the situation is more difficult because $\left\|\nabla f(\mathbf{x}_{\text{prev}(k,m)}; \xi^m_{\text{prev}(k,m)})\right\|$ could be large. Our solution to this issue is to introduce a new *delay-adaptive* stepsize schedule $\gamma_k \sim 1/\tau(k)$, which we show allows for sufficient control over $\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|$. Similar stepsizes have been considered by [WMFJ22] for the PIAG algorithm, while the stepsizes for Asynchronous SGD used in previous analyses typically scale with $1/\tau_{\max}$ [SK20, e.g.,]. Thus, our analysis shows that we can get away with a more aggressive stepsize to get better rates. However, our stepsize choice could be problematic if it were correlated with the noise in the stochastic gradients because our proofs involve the step:

$$\mathbb{E}_{\xi^{m_k}_{\text{prev}(k,m_k)} \sim \mathcal{D}}\left[\gamma_k \nabla f(\mathbf{x}_{\text{prev}(k,m_k)}; \xi^{m_k}_{\text{prev}(k,m_k)})\right] = \gamma_k \nabla F(\mathbf{x}_{\text{prev}(k,m_k)}).$$

Therefore, we introduce the following assumption about the relationship between the delays and data.

Assumption 3.4.1. *The stochastic sequences* $(\xi_1, \xi_2, \ldots)$ *and* $(\tau(0), \tau(1), \ldots)$ *are independent.*

This assumption holds, for example, when it takes a fixed amount of computation to evaluate any stochastic gradient, and the combination of the (potentially heterogeneous) computational throughput on the different workers and network latency gives rise to the delays. However, this can fail to hold, for instance, when training a model with variable-length inputs, in which case the delays will probably depend on the length of the input sequences, so it will likely also be related to the gradient noise. The next Theorem shows our convergence guarantees under Assumption 3.4.1 for Asynchronous SGD with novel delay-adaptive stepsizes scaling with $1/\tau(k)$:

Theorem 3.2. *Suppose* F *is* L*-smooth, that Assumption 3.4.1 holds, that* $B \geq \mathbb{E}\|\mathbf{x}_0 - \mathbf{x}^*\|^2$ *for some minimizer* $\mathbf{x}^*$*, and* $\Delta \geq \mathbb{E}F(\mathbf{x}_0) - F^*$*. Then there exist numerical constants* c_1, c_2, c_3, c_4 *such that:*

1. *For convex* F*,* $K \geq M$ *and* $\gamma_k = \min\left\{\frac{1}{4L\tau(k)}, \frac{1}{4ML}, \frac{B}{\sigma\sqrt{K}}\right\}$ *Algorithm 3.1 ensures*

$$\mathbb{E}\left[F(\tilde{\mathbf{x}}_K) - F^*\right] \leq c_1 \cdot \left(\frac{MLB^2}{K} + \frac{\sigma B}{\sqrt{K}}\right),$$

   where $\tilde{x}_K$ *is a weighted average*[3] *of* $\mathbf{x}_1, \ldots, \mathbf{x}_K$*.*

2. *For* μ*-strongly convex* F*,* $K \geq 3M$*, and* $\gamma_k = \min\left\{\frac{\exp\left(\frac{-\mu\tau(k)}{4ML}\right)}{4L\tau(k)}, \frac{1}{8ML}, \frac{504\ln\left(e + \frac{\mu^2K^2B^2}{\sigma^2}\right)}{\mu K}\right\}$

   Algorithm 3.1 ensures

$$\mathbb{E}[F(\tilde{\mathbf{x}}_K) - F^*] \leq c_2 \cdot \left(MLB^2 \exp\left(-\frac{c_3 K\mu}{ML}\right) + \frac{\sigma^2}{\mu K}\log\left(e + \frac{\mu^2K^2B^2}{\sigma^2}\right)\right),$$

   where $\tilde{\mathbf{x}}_K$ *is a weighted average of* $\mathbf{x}_1, \ldots, \mathbf{x}_K$ *defined.*

[3] The weight on each iterate $\mathbf{x}_k$ is proportional to $\hat{\gamma}_k$, which depends on the eventual delay of $\nabla f(\mathbf{x}_k; \xi_k^{m_k})$, and is thus not yet known at iteration k. However, on line 4 of Algorithm 3.1, the worker could simply return both its gradient and the point at which it was evaluated, so that the term $\hat{\gamma}_k \mathbf{x}_k$ can simply be added at iteration $\text{next}(k+1, m_k)$ rather than at iteration k, so there is not need to store all of the previous iterates.

3. *For non-convex F, $K \geq M$, and $\gamma_k = \min\left\{\frac{1}{4L\tau(k)}, \frac{1}{2ML}, \sqrt{\frac{\Delta}{KL\sigma^2}}\right\}$ Algorithm 3.1 ensures*

$$\mathbb{E}\left[\|\nabla F(\tilde{\mathbf{x}}_K)\|^2\right] \leq c_4 \cdot \left(\frac{ML\Delta}{K} + \sqrt{\frac{L\Delta\sigma^2}{K}}\right),$$

where $\tilde{\mathbf{x}}_K$ is randomly chosen from $\mathbf{x}_1, \ldots, \mathbf{x}_K$ according to (3.11).

We provide proof of Theorem 3.2.3 in Section 3.D (non-convex setting). The convex and strongly convex proofs follow the same lines, and we refer the reader to the appendix of [MBEW22] for convex proofs. To understand the implications of Theorem 3.2, we recall the guarantees for R steps of Minibatch SGD using minibatches of size M in the setting of Theorem 3.2, which are (ignoring all constants) [NY83, GL13]: $\mathbb{E}[F(\mathbf{x}_{\text{Mini}}) - F^*] \leq \frac{LB^2}{R} + \frac{\sigma B}{\sqrt{RM}}$ in the convex setting, $\mathbb{E}[F(\mathbf{x}_{\text{Mini}}) - F^*] \leq LB^2 \exp(-\frac{\mu R}{L}) + \frac{\sigma^2}{\mu RM}$ in the strongly convex setting, and $\mathbb{E}\|\nabla F(\mathbf{x}_{\text{Mini}})\|^2 \leq \frac{L\Delta}{R} + \sqrt{\frac{L\Delta\sigma^2}{RM}}$ in the non-convex setting. Comparing these to the guarantees for Asynchronous SGD in Theorem 3.2.1–3, we see that up to constants (and logarithmic factors in the strongly convex case), our results match the guarantees of $R = K/M$ steps of Minibatch SGD with minibatch size M. As discussed in Section 3.1.2, each update of Minibatch SGD takes the time needed by the *slowest* machine, meaning that in any given amount of time, Asynchronous SGD will complete at least M times more updates than Minibatch would. Therefore, the guarantees in Theorem 3.2 imply *strictly better* performance for Algorithm 3.1 than for Minibatch SGD in terms of guaranteed error after a fixed amount of time, as illustrated in Table 3.2. Figure 3.1 depicts a simple experiment demonstrating this phenomenon in practice.

The first "optimization" terms in our guarantees match K/M steps of exact gradient descent, completely irrespective of the delays. In the convex cases, it is likely that these could be "accelerated" to scale with $(K/M)^{-2}$ or $\exp(-\frac{K\sqrt{\mu}}{M\sqrt{L}})$, but at the expense of a more complex algorithm and analysis. However, by analogy to existing lower bounds [WWS+18, CDHS17] we conjecture that the optimization terms in Theorem 3.2.1–2 are the best "unaccelerated" rate we could hope for, and that the optimization term in Theorem 3.2.3 is optimal. Moreover, despite the delays, the second "statistical" terms in our guarantees are minimax optimal amongst all algorithms that use K stochastic gradients [NY83, ACD+22]. So, when the statistical terms dominate the rate, our guarantees are unimprovable in the worst case, and in fact they even match what could be guaranteed by K steps of SGD without delays. Furthermore, since the optimization terms decrease faster with K, in a realistic scenario where $M \ll K$, the statistical term will eventually dominate the rate and our algorithm will have optimal performance with no penalty from the delayed gradients at all.

3.5. Heterogeneous data setting

Finally, we extend our results to the heterogeneous data setting, where each machine m possesses its own local data distribution $\mathcal{D}_m$, giving rise to a local objective F_m. The optimization problem in the heterogeneous setting is:

$$\min_{\mathbf{x}\in\mathbb{R}^d}\left\{F(x) = \frac{1}{M}\sum_{m=1}^{M} F_m(\mathbf{x})\right\}, \quad \text{where} \quad F_m(\mathbf{x}) = \mathbb{E}_{\xi\sim\mathcal{D}_m}[f_m(\mathbf{x};\xi)]\,. \tag{3.7}$$

In this setting, we define Asynchronous SGD the same way, with worker m having access to the stochastic gradients $\nabla f_m(\cdot;\xi)$ and updates taking the form:

$$\mathbf{x}_k = \mathbf{x}_{k-1} - \gamma_k \nabla f_{m_k}(\mathbf{x}_{\text{prev}(k,m_k)}; \xi^{m_k}_{\text{prev}(k,m_k)})\,. \tag{3.8}$$

It is generally impossible to show that Asynchronous SGD will work well in this setting. Specifically, if $F_2 = \cdots = F_M = 0$, then the time needed to optimize F is entirely dependent

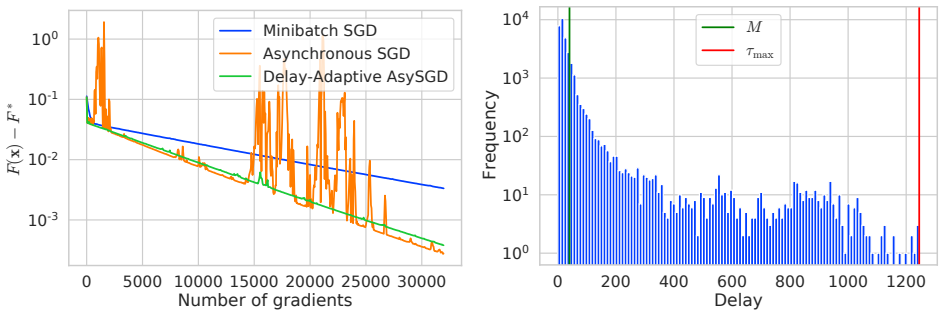


Figure 3.1 – *We ran an experiment on a simple least-squares problem with random data and tuned all stepsizes. On the left, we see that after a fixed number of gradients are used, Asynchronous SGD is slightly better than Minibatch SGD but it is also quite unstable, whereas with delay-adaptive stepsizes it is both fast and stable. On the right, we can see the distribution of the delays (note that y-axis is in log scale). Additional details about the experiment can be found in Appendix 3.A. Credits to Konstantin Mishchenko for the figure.*

on the speed of the first worker. Moreover, with arbitrary delays, we might only get a single gradient update from the first worker, leaving us no hope of any useful guarantee. To avoid this issue and to be able to apply our analysis, we will require that the gradient dissimilarities between the local functions are bounded and that the identity of the worker used in iteration k is independent of the gradient noise:

Assumption 3.5.1. *There exists $\zeta \geq 0$ such that $\|\nabla F_m(\mathbf{x}) - \nabla F(\mathbf{x})\|^2 \leq \zeta^2$ for all m and $\mathbf{x} \in \mathbb{R}^d$, $\mathbb{E}[\mathbf{g}_k | \mathbf{x}_k, m_k] = \nabla F_{m_k}(\mathbf{x}_k)$, and $\mathbb{E}\left\|\|\mathbf{g}_k - \nabla F_{m_k}(\mathbf{x}_k)\|^2 | \mathbf{x}_k, m_k\right\| \leq \sigma^2$.*

Equipped with the new assumption, we can provide an analogue of Theorem 3.2 for problem (3.7):

Theorem 3.3. *In the setting of Theorem 3.2 with the addition of Assumption 3.5.1, for $\gamma_k = \min\left\{\frac{1}{8L\tau(k)}, \frac{1}{4ML}, \sqrt{\Delta/(KL\sigma^2)}\right\}$, the updates described in (3.8) ensure for a numerical constant c:*

$$\mathbb{E}\left[\|\nabla F(\tilde{\mathbf{x}}_K)\|^2\right] \leq c \cdot \left(\frac{ML\Delta}{K} + \sqrt{\frac{L\Delta\sigma^2}{K}} + \zeta^2\right),$$

where $\tilde{\mathbf{x}}_k$ is randomly chosen from $\mathbf{x}_1, \ldots, \mathbf{x}_K$ according to (3.14).

Comparing Theorem 3.3—which we prove in Appendix 3.D—with Theorem 3.2.3, we see that the exact same rate is achieved in the heterogeneous setting up to the additive term ζ^2. As described above, we cannot really expect good performance for arbitrarily heterogeneous losses under arbitrary delays, so some dependence on ζ^2 is unavoidable. Moreover, in many natural settings, ζ^2 can be quite small. For instance, when heterogeneity arises because a large i.i.d. dataset is partitioned across the M machines, the degree of heterogeneity ζ^2 will be inversely proportional to the number of samples assigned to each worker. So although Asynchronous SGD is not well-suited to problem (3.7) in the heterogeneous setting when the delays can be arbitrarily large, at least under Assumption 3.5.1, it does not totally break down or diverge.

While Assumption 3.5.1 used in Theorem 3.3 seems unavoidable for studying Asynchronous SGD, it can be circumvented when studying other asynchronous methods. In particular, [KSJ22] proposed to sample a random worker at each iteration k and add the current points $\mathbf{x}_k$ to its list of points for computing the gradient. Under an extra assumption on the delay pattern, [KSJ22] proved convergence of this method to a point with zero gradient, without extra errors. Unfortunately, unlike Asynchronous SGD, their method would not have the speedup shown in Table 3.2, since after a large number of iterations K, all workers

would be expected to compute roughly $\frac{K}{M}$ gradients. Since this puts the same load on the slowest worker as Minibatch SGD, the algorithm proposed by [KSJ22] is useful only when there is no worker that is fundamentally slower than the others. Our theory, in turn, has an extra ζ^2 error term, but does not require waiting for the slow workers.

Conclusion

This chapter studied Asynchronous SGD via a virtual-iterate analysis: we prove that in a vast variety of settings – convex/Lipschitz, non-convex/Lipschitz/smooth, convex/smooth, strongly-convex/smooth and non-convex/smooth – the convergence guarantees of Asynchronous SGD improve over that of Minibatch SGD, *irrespectively of the delay sequence*. We obtained these results by leveraging delay-adaptive stepsizes and proving that Asynchronous SGD is only slowed down by the number of stochastic gradients being computed, rather than the longest delay. We also extend one of these results to the case of heterogeneous data and show that Asynchronous SGD can converge to an approximate stationary point with the error controlled by a data dissimilarity constant.

Appendix of Chapter 3

3.A. Experimental details and credits for Figure 3.1

To showcase how our theory matches the numerical performance of Asynchronous SGD, we run experiments on a simple quadratic objective with random data. We run our experiments on a single node with 48 logical cores and set $M = 40$. We use the Ray package [MNW+18] to parallelize the execution and follow the official documentation for the implementation[4] of asynchronous training. All delays appearing in the runs are not simulated and come from the execution on CPUs. The implementation and the figure have been done by Konstantin Mishchenko.

3.B. Technical lemmas

Lemma 3.B.1. *Under Assumption 3.4.1 and the σ^2 variance bound, if γ_k depends only on k and $\tau(k)$ for each k, then for all $k \geq 1$,*

$$\mathbb{E}\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 \leq 2\mathbb{E}\left[\sum_{m\in[M]\setminus\{m_k\}} \gamma^2_{\text{next}(k,m)}\left[\sigma^2 + (M-1)\left\|\nabla F(\mathbf{x}_{\text{prev}(k,m)})\right\|^2\right]\right].$$

Proof. By Lemma 3.2.1, we have

$$\begin{aligned}
\mathbb{E}\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 &= \mathbb{E}\left\|\sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)}\nabla f(\mathbf{x}_{\text{prev}(k,m)}; \xi^m_{\text{prev}(k,m)})\right\|^2 \\
&\leq 2\mathbb{E}\left\|\sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)}\nabla F(\mathbf{x}_{\text{prev}(k,m)})\right\|^2 \\
&\quad + 2\mathbb{E}\left\|\sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)}(\nabla f(\mathbf{x}_{\text{prev}(k,m)}; \xi^m_{\text{prev}(k,m)}) - \nabla F(\mathbf{x}_{\text{prev}(k,m)}))\right\|^2.
\end{aligned}$$

Notice that we had to use Young's inequality to separate expectations from the variance terms since the variance in the vectors is not independent. The first term can be bounded using Young's inequality as follows,

$$\mathbb{E}\left\|\sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)}\nabla F(\mathbf{x}_{\text{prev}(k,m)})\right\|^2 \leq \sum_{m\in[M]} \gamma^2_{\text{next}(k,m)}\|\nabla F(\mathbf{x}_{\text{prev}(k,m)})\|^2.$$

To bound the second term, assume without loss of generality that $\text{prev}(k,1) < \text{prev}(k,2) <$

[4] https://docs.ray.io/en/latest/ray-core/examples/plot_parameter_server.html#asynchronous-parameter-server-training

$\cdots < \mathrm{prev}(k, M)$. In addition, denote

$$\theta_m = \gamma_{\mathrm{next}(k,m)}(\nabla f(\mathbf{x}_{\mathrm{prev}(k,m)}; \xi^m_{\mathrm{prev}(k,m)}) - \nabla F(\mathbf{x}_{\mathrm{prev}(k,m)})).$$

Then, we have for any m

$$\mathbb{E}\left[\|\theta_m\|^2\right] = \gamma^2_{\mathrm{next}(k,m)}\mathbb{E}\|\nabla f(\mathbf{x}_{\mathrm{prev}(k,m)}; \xi^m_{\mathrm{prev}(k,m)}) - \nabla F(\mathbf{x}_{\mathrm{prev}(k,m)})\|^2 \leq \gamma^2_{\mathrm{next}(k,m)}\sigma^2.$$

Moreover, for any $m \in [1, M-1]$, the stochastic gradient of worker $m+1$ has conditional expectation

$$\mathbb{E}\left[\nabla f(\mathbf{x}_{\mathrm{prev}(k,m+1)}; \xi^{m+1}_{\mathrm{prev}(k,m+1)}) \mid \nabla f(\mathbf{x}_{\mathrm{prev}(k,m)}; \xi^m_{\mathrm{prev}(k,m)})\right] = \nabla F(\mathbf{x}_{\mathrm{prev}(k,m+1)}),$$

so $\mathbb{E}\left[\theta_{m+1} \mid \theta_1, \ldots, \theta_m\right] = 0$. This allows us to obtain by induction,

$$\begin{aligned}
\mathbb{E}\left[\|\sum_{j=1}^{m+1}\theta_j\|^2\right] &= \mathbb{E}\left[\|\sum_{j=1}^{m}\theta_j\|^2 + 2\langle\sum_{j=1}^{m}\theta_j, \theta_{m+1}\rangle + \|\theta_{m+1}\|^2\right] \\
&\leq \mathbb{E}\left[\|\sum_{j=1}^{m}\theta_j\|^2 + 2\langle\sum_{j=1}^{m}\theta_j, \theta_{m+1}\rangle\right] + \gamma^2_{\mathrm{next}(k,m+1)}\sigma^2 \\
&\leq \mathbb{E}\left[\sum_{j=1}^{m}\hat{\gamma}^2_{t-\tau_j}\sigma^2 + 2\langle\sum_{j=1}^{m}\theta_j, \theta_{m+1}\rangle\right] + \gamma^2_{\mathrm{next}(k,m+1)}\sigma^2 \\
&= \mathbb{E}\left[2\langle\sum_{j=1}^{m}\theta_j, \theta_{m+1}\rangle\right] + \sum_{j=1}^{m+1}\gamma^2_{\mathrm{next}(k,j)}\sigma^2.
\end{aligned}$$

The remaining scalar product is, in fact, equal to zero. Indeed, by the tower property of expectation,

$$\begin{aligned}
\mathbb{E}\left[\langle\sum_{j=1}^{m}\theta_j, \theta_{m+1}\rangle\right] &= \mathbb{E}\left[\mathbb{E}\left[\langle\sum_{j=1}^{m}\theta_j, \theta_{m+1}\rangle \mid \theta_1, \ldots, \theta_m\right]\right] \\
&= \mathbb{E}\left[\langle\sum_{j=1}^{m}\theta_j, \mathbb{E}\left[\theta_{m+1} \mid \theta_1, \ldots, \theta_m\right]\rangle\right] \\
&= 0.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\mathbb{E}\left\|\sum_{m\in[M]\setminus\{m_k\}}\gamma_{\mathrm{next}(k,m)}(\nabla f(\mathbf{x}_{\mathrm{prev}(k,m)}; \xi^m_{\mathrm{prev}(k,m)}) - \nabla F(\mathbf{x}_{\mathrm{prev}(k,m)}))\right\|^2 \\
\leq \sum_{m\in[M]\setminus\{m_k\}}\gamma^2_{\mathrm{next}(k,m)}\sigma^2.
\end{aligned}$$

□

Lemma 3.B.1 is very useful whenever stochastic gradients are not guaranteed to be bounded. If they were bounded, we could immediately show that $\mathbb{E}\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2$ is small regardless of the delays, as was done in the proof of Theorem 1, see equation (3.5). For non-Lipschitz losses, however, $\mathbb{E}\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2$ is not guaranteed to be finite, and Lemma 3.B.1

is required to show that $\hat{\mathbf{x}}_k$ and $\mathbf{x}_k$ stay sufficiently close to each other.

The next lemma is a general statement about sequences with delays, which we will use to bound various error terms in our proofs.

Lemma 3.B.2. *For any positive sequences $\{a_k\}$, $\{b_k\}$, and $\{c_k\}$, it holds*

$$\sum_{k=1}^{K} \sum_{m\in[M]\setminus\{m_k\}} a_k b_{\mathrm{prev}(k,m)} c_{\mathrm{next}(k,m)} = b_0 \sum_{m=1}^{M} c_{\mathrm{next}(1,m)} \sum_{j=1}^{\min\{\mathrm{next}(1,m)-1,\,K\}} a_j + \sum_{k=1}^{K-1} b_k c_{\mathrm{next}(k+1,m_k)} \sum_{j=k+1}^{\min\{\mathrm{next}(k+1,m_k)-1,\,K\}} a_j \,.$$

Proof. The proof follows simply by rewriting the sums several times while manipulating the definitions of prev and next:

$$\begin{aligned}
&\sum_{k=1}^{K} \sum_{m\in[M]\setminus\{m_k\}} a_k b_{\mathrm{prev}(k,m)} c_{\mathrm{next}(k,m)} \\
&\quad= \sum_{j=0}^{K-1} \sum_{k=1}^{K} \sum_{m=1}^{M} a_k b_j c_{\mathrm{next}(k,m)} \mathbf{1}_{\{m\neq m_k\}} \mathbf{1}_{\{j=\mathrm{prev}(k,m)\}} \\
&\quad= b_0 \sum_{k=1}^{K} \sum_{m=1}^{M} a_k c_{\mathrm{next}(k,m)} \mathbf{1}_{\{m\neq m_k\}} \mathbf{1}_{\{0=\mathrm{prev}(k,m)\}} \\
&\qquad\qquad + \sum_{j=1}^{K-1} \sum_{k=1}^{K} \sum_{m=1}^{M} a_k b_j c_{\mathrm{next}(k,m)} \mathbf{1}_{\{m\neq m_k\}} \mathbf{1}_{\{j=\mathrm{prev}(k,m)\}} \\
&\quad= b_0 \sum_{m=1}^{M} c_{\mathrm{next}(1,m)} \sum_{k=1}^{\min\{\mathrm{next}(1,m),\,K\}} a_k \mathbf{1}_{\{m\neq m_k\}} \\
&\qquad\qquad + \sum_{j=1}^{K-1} b_j c_{\mathrm{next}(j+1,m_j)} \sum_{k=j+1}^{\min\{\mathrm{next}(j+1,m_j),\,K\}} a_k \mathbf{1}_{\{m_j\neq m_k\}} \\
&\quad= b_0 \sum_{m=1}^{M} c_{\mathrm{next}(1,m)} \sum_{k=1}^{\min\{\mathrm{next}(1,m)-1,\,K\}} a_k + \sum_{j=1}^{K-1} b_j c_{\mathrm{next}(j+1,m_j)} \sum_{k=j+1}^{\min\{\mathrm{next}(j+1,m_j)-1,\,K\}} a_k,
\end{aligned}$$

which establishes the claim after exchanging subscripts. □

3.C. Proof of Theorem 3.2.3 (non-convex-smooth case)

Throughout this proof, we will refer frequently to $\hat{\gamma}_1, \ldots, \hat{\gamma}_K$, the stepsizes corresponding to the stochastic gradients $\nabla f(\mathbf{x}_1, \xi_1^{m_1}), \ldots, \nabla f(\mathbf{x}_K, \xi_K^{m_K})$. However, some of these gradients will not be available when the algorithm ends after K updates—in particular, precisely $M-1$ of them are in the process of being calculated when the algorithm terminates. Since those gradients are never used for updates, the corresponding stepsize $\hat{\gamma}_k$ seems, in some sense, unneeded. However, our algorithm's output weights each iterate $\mathbf{x}_k$ by a term involving $\hat{\gamma}_k$, whether or not the gradient $\nabla f(\mathbf{x}_k, \xi_k^{m_k})$ becomes available before the algorithm finishes. For this reason, in order to make the stepsize $\hat{\gamma}_k$ well-defined, we specify $\tau(k)$ for $k \geq K$, upon which $\hat{\gamma}_1, \ldots, \hat{\gamma}_K$ might depend, according to $\tau(k) = \max\{1, \min\{k, K\} - \mathrm{prev}(k, m_k)\}$. This is essentially equivalent to just not incrementing the iteration counter once it reaches K, although we note that all of the stepsizes $\hat{\gamma}_1, \ldots, \hat{\gamma}_K$ can be calculated by the time of the

Kth update, without needing to wait for the $M-1$ in-progress gradients.

Lemma 3.C.1. *In the setting of Theorem 3.2.3, with $\gamma_{\max}$ defined as in (3.10)*

$$\mathbb{E}\left[\frac{L^2}{2}\sum_{k=1}^{K}\hat{\gamma}_k\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|^2\right]\leq\mathbb{E}\left[\frac{\Delta}{8}+\frac{L\sigma^2\gamma_{\max}}{4}\left(M\gamma_{\max}+\sum_{k=1}^{K}\hat{\gamma}_k\right)+\frac{1}{8}\sum_{k=1}^{K}\hat{\gamma}_k\|\nabla F(\mathbf{x}_k)\|^2\right].$$

Proof. We start using Lemma 3.B.1 and then Lemma 3.B.2 with $a_k=\hat{\gamma}_k$, $b_k=\sigma^2+(M-1)\|\nabla F(\mathbf{x}_k)\|^2$, and $c_k=\gamma_k^2$:

$$\begin{aligned}
&\mathbb{E}\left[\frac{L^2}{2}\sum_{k=1}^{K}\hat{\gamma}_k\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|^2\right]\\
&\quad\leq L^2\mathbb{E}\left[\sum_{k=1}^{K}\hat{\gamma}_k\sum_{m\in[M]\setminus\{m_k\}}\gamma^2_{\text{next}(k,m)}\left[\sigma^2+(M-1)\left\|\nabla F(\mathbf{x}_{\text{prev}(k,m)})\right\|^2\right]\right]\\
&\quad\leq L^2\gamma_{\max}\mathbb{E}\left[\left\{\sigma^2+(M-1)\|\nabla F(\mathbf{x}_0)\|^2\right\}\sum_{m=1}^{M}\gamma^2_{\text{next}(1,m)}\min\left\{\text{next}(1,m)-1,\,K\right\}\right.\\
&\qquad\left.+\sum_{k=1}^{K-1}\left(\sigma^2+(M-1)\|\nabla F(\mathbf{x}_k)\|^2\right)\hat{\gamma}_k^2(\min\left\{\text{next}(k+1,m_k)-1,\,K\right\}-k)\right].
\end{aligned}$$

From here, note that our choice of stepsize

$$\gamma_k\leq\frac{1}{4L\tau(k)}$$

implies

$$\begin{aligned}
\gamma_{\text{next}(1,m)}\times\min\left\{\text{next}(1,m)-1,\,K\right\}&\leq\frac{1}{4L},\\
\hat{\gamma}_k\times(\min\left\{\text{next}(k+1,m_k)-1,\,K\right\}-k)&\leq\frac{1}{4L}.
\end{aligned}$$

Plugging these two inequalities into our previous bound, we obtain

$$\begin{aligned}
&\mathbb{E}\left[\frac{L^2}{2}\sum_{k=1}^{K}\hat{\gamma}_k\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|^2\right]\\
&\quad\leq\frac{L\gamma_{\max}}{4}\mathbb{E}\left[\left(\sigma^2+M\|\nabla F(\mathbf{x}_0)\|^2\right)\sum_{m=1}^{M}\gamma_{\text{next}(1,m)}+\sum_{k=1}^{K-1}\left(\sigma^2+M\|\nabla F(\mathbf{x}_k)\|^2\right)\hat{\gamma}_k\right]\\
&\quad\leq\mathbb{E}\left[\frac{ML\gamma_{\max}^2}{4}\left(\sigma^2+M\|\nabla F(\mathbf{x}_0)\|^2\right)+\frac{L\sigma^2\gamma_{\max}}{4}\sum_{k=1}^{K}\hat{\gamma}_k+\frac{ML\gamma_{\max}}{4}\sum_{k=1}^{K}\hat{\gamma}_k\|\nabla F(\mathbf{x}_k)\|^2\right].
\end{aligned}$$

Using the fact that $\gamma_{\max}\leq 1/(2ML)$ and $\|\nabla F(x_0)\|^2\leq 2L\Delta$ completes the proof. □

Lemma 3.C.2. *In the setting of Theorem 3.2.3, with $\gamma_{\max}$ defined as in (3.10)*

$$\sum_{k=1}^{K}\hat{\gamma}_k\geq\frac{K\gamma_{\max}}{9}.$$

Proof. First, we note that if $\tau(k) \leq 3M$, then

$$\gamma_k = \min\left\{\frac{1}{6L}, \frac{1}{2ML}, \sqrt{\frac{\Delta}{KL\sigma^2}}\right\} \geq \frac{\gamma_{\max}}{3}.$$

We begin with the observation that

$$\sum_{k=1}^{K-1} \tau(k, m_k) + \sum_{m=1}^{M} \tau(K, m) \leq \sum_{k=1}^{K}\sum_{m=1}^{M} \tau(k, m) \leq KM. \tag{3.9}$$

This implies that at most $\lfloor K/3 \rfloor$ of the terms on the left hand side can be larger than $3M$. Furthermore, since the first $3M$ gradients must have delay less than $3M$, the number of terms with delay greater than $3M$ is no larger than $\min\{K/3, \max\{K - 3M, 0\}\}$.

From here, we rewrite:

$$\begin{aligned}
\sum_{k=1}^{K} \hat{\gamma}_k &= \sum_{k=1}^{K-1} \gamma_k \mathbf{1}_{\{\text{prev}(k,m_k)>0\}} + \sum_{m=1}^{M} \gamma_{\text{next}(K,m)} \mathbf{1}_{\{\text{prev}(K,m)>0\}} \\
&\geq \sum_{k=1}^{K-1} \gamma_k \mathbf{1}_{\{\text{prev}(k,m_k)>0\}} \mathbf{1}_{\{\tau(k,m_k)\leq 3M\}} + \sum_{m=1}^{M} \gamma_{\text{next}(K,m)} \mathbf{1}_{\{\text{prev}(K,m)>0\}} \mathbf{1}_{\{\tau(K,m)\leq 3M\}} \\
&\geq \frac{\gamma_{\max}}{3}\left(\sum_{k=1}^{K-1} \mathbf{1}_{\{\text{prev}(k,m_k)>0\}} \mathbf{1}_{\{\tau(k,m_k)\leq 3M\}} + \sum_{m=1}^{M} \mathbf{1}_{\{\text{prev}(K,m)>0\}} \mathbf{1}_{\{\tau(K,m)\leq 3M\}}\right) \\
&\geq \frac{\gamma_{\max}}{3}\left(K + M - 1 - \left(\min\left\{\frac{K}{3}, \max\{K - 3M, 0\}\right\} + M\right)\right) \\
&= \frac{\gamma_{\max}}{3} \max\left\{\frac{2K}{3} - 1, \min\{3M - 1, K - 1\}\right\} \\
&\geq \frac{\gamma_{\max}}{3} \min\left\{\frac{K}{3}, K - 1\right\}.
\end{aligned}$$

For $K \geq 2$, the Lemma follows directly. For $K = 1$, we also have $M = 1$ so all of the delays are one, and $\sum_{k=1}^{K} \hat{\gamma}_k = \hat{\gamma}_1 = \min\left\{\gamma_{\max}, \frac{1}{2L}\right\} \geq \frac{K\gamma_{\max}}{9}$. Therefore, this inequality holds either way, completing the proof. □

Proof of Theorem 3.2.3. In this proof, we will use $\gamma_{\max}$ to denote the $\tau(k)$-independent terms in the definition of the stepsize, i.e.,

$$\gamma_{\max} = \min\left\{\frac{1}{2ML}, \sqrt{\frac{\Delta}{KL\sigma^2}}\right\}. \tag{3.10}$$

Next, we use the L-smoothness of F, the definition of $\hat{\mathbf{x}}_{k+1}$ from (3.3), and Assumption 3.4.1:

$$\begin{aligned}
\mathbb{E}F(\hat{\mathbf{x}}_{k+1}) &\leq \mathbb{E}\left[F(\hat{\mathbf{x}}_k) + \langle \nabla F(\hat{\mathbf{x}}_k), \hat{\mathbf{x}}_{k+1} - \hat{\mathbf{x}}_k\rangle + \frac{L}{2}\|\hat{\mathbf{x}}_{k+1} - \hat{\mathbf{x}}_k\|^2\right] \\
&= \mathbb{E}\left[F(\hat{\mathbf{x}}_k) - \hat{\gamma}_k\langle \nabla F(\hat{\mathbf{x}}_k)\nabla F(\mathbf{x}_k)\rangle + \frac{L\hat{\gamma}_k^2}{2}\left\|\nabla f(\mathbf{x}_k; \xi_k^{m_k})\right\|^2\right] \\
&\leq \mathbb{E}\left[F(\hat{\mathbf{x}}_k) + \left(\frac{L\hat{\gamma}_k^2}{2} - \frac{\hat{\gamma}_k}{2}\right)\|\nabla F(\mathbf{x}_k)\|^2 + \frac{\hat{\gamma}_k}{2}\|\nabla F(\hat{\mathbf{x}}_k) - \nabla F(\mathbf{x}_k)\|^2 + \frac{L\sigma^2\hat{\gamma}_k^2}{2}\right] \\
&\leq \mathbb{E}\left[F(\hat{\mathbf{x}}_k) + \left(\frac{L\hat{\gamma}_k^2}{2} - \frac{\hat{\gamma}_k}{2}\right)\|\nabla F(\mathbf{x}_k)\|^2 + \frac{L^2\hat{\gamma}_k}{2}\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 + \frac{L\sigma^2\hat{\gamma}_k^2}{2}\right].
\end{aligned}$$

Since $\gamma_k \le \frac{1}{2L}$ for all k, this means

$$\mathbb{E}[F(\hat{\mathbf{x}}_{k+1}) - F(\hat{\mathbf{x}}_k)] \le \mathbb{E}\left[-\frac{\hat{\gamma}_k}{4}\|\nabla F(\mathbf{x}_k)\|^2 + \frac{L^2\hat{\gamma}_k}{2}\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 + \frac{L\sigma^2\hat{\gamma}_k^2}{2}\right].$$

Rearranging and summing over k, this means

$$\begin{aligned}\frac{1}{4}\mathbb{E}\left[\sum_{k=1}^K \hat{\gamma}_k\|\nabla F(\mathbf{x}_k)\|^2\right] &\le \mathbb{E}\left[F(\hat{\mathbf{x}}_1) - F(\hat{\mathbf{x}}_{K+1}) + \frac{L^2}{2}\sum_{k=1}^K \hat{\gamma}_k\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 + \frac{L\sigma^2}{2}\sum_{k=1}^K \hat{\gamma}_k^2\right] \\ &\le \mathbb{E}\left[\Delta + \frac{L^2}{2}\sum_{k=1}^K \hat{\gamma}_k\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 + \frac{L\sigma^2}{2}\sum_{k=1}^K \hat{\gamma}_k^2\right].\end{aligned}$$

Applying Lemma 3.C.1 and rearranging, this gives

$$\mathbb{E}\left[\sum_{k=1}^K \hat{\gamma}_k\|\nabla F(\mathbf{x}_k)\|^2\right] \le \mathbb{E}\left[9\Delta + 2ML\sigma^2\gamma_{\max}^2 + 6L\sigma^2\gamma_{\max}\sum_{k=1}^K \hat{\gamma}_k\right].$$

Therefore, if we choose an output vector $\tilde{\mathbf{x}}_K$ with

$$\mathbb{P}(\tilde{\mathbf{x}}_K = \mathbf{x}_k) \propto \hat{\gamma}_k \quad \forall k \in [K], \tag{3.11}$$

then by Lemma 3.C.2,

$$\begin{aligned}\mathbb{E}\left[\|\nabla F(\tilde{\mathbf{x}}_K)\|^2\right] &\le \mathbb{E}\left[\frac{9\Delta + 2ML\sigma^2\gamma_{\max}^2}{\sum_{k=1}^K \hat{\gamma}_k} + 6L\sigma^2\gamma_{\max}\right] \\ &\le \mathbb{E}\left[\frac{81\Delta + 18ML\sigma^2\gamma_{\max}^2}{K\gamma_{\max}} + 6L\sigma^2\gamma_{\max}\right].\end{aligned}$$

Substituting $\gamma_{\max}$ from (3.10) completes the proof. □

3.D. The heterogeneous data setting: proof of Theorem 3.3

The analysis in this section is not too different from that before. The main idea here is to refine the upper bound on the distance between the virtual and actual iterates, which can be done using our assumption on bounded data heterogeneity.

Lemma 3.D.1. *In the setting of Theorem 3.3, for any $k \ge 1$*

$$\mathbb{E}\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 \le 2\mathbb{E}\left[\sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)}^2\left[\sigma^2 + 2(M-1)\left(\left\|\nabla F(\mathbf{x}_{\text{prev}(k,m)})\right\|^2 + \zeta^2\right)\right]\right].$$

Proof. The argument below is nearly identical to the proof of Lemma 3.B.1. We start with an analogue of Lemma 3.2.1:

$$\mathbf{x}_k - \hat{\mathbf{x}}_k = \sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)}\nabla f_m(\mathbf{x}_{\text{prev}(k,m)}; \xi^m_{\text{prev}(k,m)}),$$

which follows from exactly the same argument. We then use Assumptions 3.4.1 and 3.5.1 in

the same way as in the proof of Lemma 3.B.1:

$$\begin{aligned}
\mathbb{E}\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 &= \mathbb{E}\left\|\sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)} \nabla f_m(\mathbf{x}_{\text{prev}(k,m)}; \xi^m_{\text{prev}(k,m)})\right\|^2 \\
&\leq 2\mathbb{E}\left[\sigma^2 \sum_{m\in[M]\setminus\{m_k\}} \gamma^2_{\text{next}(k,m)} + \left\|\sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)} \nabla F_m(\mathbf{x}_{\text{prev}(k,m)})\right\|^2\right] \\
&\leq 2\mathbb{E}\left[\sigma^2 \sum_{m\in[M]\setminus\{m_k\}} \gamma^2_{\text{next}(k,m)} + 2\left\|\sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)} \nabla F(\mathbf{x}_{\text{prev}(k,m)})\right\|^2\right. \\
&\qquad \left. + 2\left\|\sum_{m\in[M]\setminus\{m_k\}} \gamma_{\text{next}(k,m)} \big(\nabla F_m(\mathbf{x}_{\text{prev}(k,m)}) - \nabla F(\mathbf{x}_{\text{prev}(k,m)})\big)\right\|^2\right] \\
&\leq 2\mathbb{E}\left[\sum_{m\in[M]\setminus\{m_k\}} \gamma^2_{\text{next}(k,m)} \left[\sigma^2 + 2(M-1)\left\|\nabla F(\mathbf{x}_{\text{prev}(k,m)})\right\|^2 + 2(M-1)\zeta^2\right]\right].
\end{aligned}$$

□

Lemma 3.D.2. *In the setting of Theorem 3.3, with* $\gamma_{\max}$ *defined as in* (3.12)

$$\begin{aligned}
&8L^2 \sum_{k=1}^K \hat{\gamma}_k \|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 \\
&\leq \mathbb{E}\left[\frac{\Delta}{4} + L\gamma_{\max}\sigma^2\left(M\gamma_{\max} + \sum_{k=1}^K \hat{\gamma}_k\right) + \zeta^2\left(2LM^2\gamma^2_{\max} + \frac{1}{2}\sum_{k=1}^K \hat{\gamma}_k\right) + \frac{1}{2}\sum_{k=1}^K \hat{\gamma}_k \|\nabla F(\mathbf{x}_k)\|^2\right].
\end{aligned}$$

Proof. This follows exactly the same argument as Lemma 3.C.1, just replacing that proof's invocation of Lemma 3.B.1 with Lemma 3.D.1:

$$\begin{aligned}
&\mathbb{E}\left[8L^2 \sum_{k=1}^K \hat{\gamma}_k \|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2\right] \\
&\leq 16L^2\mathbb{E}\left[\sum_{k=1}^K \hat{\gamma}_k \sum_{m\in[M]\setminus\{m_k\}} \gamma^2_{\text{next}(k,m)} \left[\sigma^2 + 2(M-1)\left(\left\|\nabla F(\mathbf{x}_{\text{prev}(k,m)})\right\|^2 + \zeta^2\right)\right]\right] \\
&\leq 16L^2\gamma_{\max}\mathbb{E}\left[\left(\sigma^2 + 2M(\|\nabla F(\mathbf{x}_0)\|^2 + \zeta^2)\right) \sum_{m=1}^M \gamma^2_{\text{next}(1,m)} \min\{\text{next}(1,m) - 1,\, K\}\right. \\
&\qquad \left. + \sum_{k=1}^{K-1} \left(\sigma^2 + 2M(\|\nabla F(\mathbf{x}_k)\|^2\right) + \zeta^2)\hat{\gamma}^2_k (\min\{\text{next}(k+1, m_k) - 1,\, K\} - k)\right].
\end{aligned}$$

The stepsize is chosen so that $\gamma_k \leq \frac{1}{8L\tau(k)}$, which implies

$$\begin{aligned}
\gamma_{\text{next}(1,m)} \min\{\text{next}(1,m) - 1,\, K\} &\leq \frac{1}{8L}, \\
\hat{\gamma}_k (\min\{\text{next}(k+1, m_k) - 1,\, K\} - k) &\leq \frac{1}{8L}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\mathbb{E}\left[8L^2\sum_{k=1}^{K}\hat{\gamma}_k\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|^2\right]\\
&\quad\le 2L\gamma_{\max}\mathbb{E}\Bigg[\left(\sigma^2+2M(\|\nabla F(\mathbf{x}_0)\|^2+\zeta^2)\right)\sum_{m=1}^{M}\gamma_{\mathrm{next}(1,m)}\\
&\qquad\qquad+\sum_{k=1}^{K-1}\left(\sigma^2+2M(\|\nabla F(\mathbf{x}_k)\|^2\right)+\zeta^2)\hat{\gamma}_k\Bigg]\\
&\quad\le L\gamma_{\max}\mathbb{E}\left[M\gamma_{\max}\left(\sigma^2+4LM^2\Delta+2M\zeta^2\right)+\sigma^2\sum_{k=1}^{K}\hat{\gamma}_k+2M\sum_{k=1}^{K}\hat{\gamma}_k(\|\nabla F(\mathbf{x}_k)\|^2+\zeta^2)\right].\\
&\quad\le\mathbb{E}\left[\frac{\Delta}{4}+L\gamma_{\max}\sigma^2\left(M\gamma_{\max}+\sum_{k=1}^{K}\hat{\gamma}_k\right)+\zeta^2\left(2LM^2\gamma_{\max}^2+\frac{1}{2}\sum_{k=1}^{K}\hat{\gamma}_k\right)+\frac{1}{2}\sum_{k=1}^{K}\hat{\gamma}_k\|\nabla F(\mathbf{x}_k)\|^2\right].
\end{aligned}$$

We used here that $\gamma_k\le 1/(8ML)$ and $\|\nabla F(\mathbf{x}_0)\|^2\le 2L\Delta$. □

Proof of Theorem 3.3. In this proof, we will use $\gamma_{\max}$ to denote the $\tau(k)$-independent terms in the definition of the stepsize, i.e.,

$$\gamma_{\max}=\min\left\{\frac{1}{4ML},\sqrt{\frac{\Delta}{KL\sigma^2}}\right\}. \tag{3.12}$$

As in the data-homogeneous setting, we define the virtual sequence $\hat{\mathbf{x}}_k$ as:

$$\begin{aligned}
\hat{\mathbf{x}}_1&=\mathbf{x}_0-\sum_{m=1}^{M}\gamma_{\mathrm{next}(1,m)}\nabla f_m(\mathbf{x}_0;\xi_0^m)\\
\hat{\mathbf{x}}_{k+1}&=\hat{\mathbf{x}}_k-\hat{\gamma}_k\nabla f_{m_k}(\mathbf{x}_k;\xi_k^{m_k}),\\
\hat{\gamma}_k&=\gamma_{\mathrm{next}(k+1,m_k)}\,.
\end{aligned} \tag{3.13}$$

Denote $\gg_k=\nabla f_{m_k}(\mathbf{x}_k;\xi_k^{m_k})$. Using the L-smoothness of F and Assumptions 3.4.1 and 3.5.1, we have:

$$\begin{aligned}
&\mathbb{E}[F(\hat{\mathbf{x}}_{k+1})-F(\hat{\mathbf{x}}_k)]\\
&\qquad\le\mathbb{E}\left[-\hat{\gamma}_k\langle\nabla F(\hat{\mathbf{x}}_k),\gg_k\rangle+\frac{\hat{\gamma}_k^2L}{2}\|\gg_k\|^2\right]\\
&\qquad\le\mathbb{E}\left[-\hat{\gamma}_k\langle\nabla F(\hat{\mathbf{x}}_k),\nabla F_{m_k}(\mathbf{x}_k)\rangle+\frac{\hat{\gamma}_k^2L}{2}(\|\nabla F_{m_k}(\mathbf{x}_k)\|^2+\sigma^2)\right]\\
&\qquad\le\mathbb{E}\left[-\frac{\hat{\gamma}_k}{4}\|\nabla F_{m_k}(\mathbf{x}_k)\|^2+\frac{\hat{\gamma}_k}{2}\|\nabla F(\hat{\mathbf{x}}_k)-\nabla F_{m_k}(\mathbf{x}_k)\|^2+\frac{\hat{\gamma}_k^2L\sigma^2}{2}\right]\\
&\qquad\le\mathbb{E}\left[-\frac{\hat{\gamma}_k}{4}\|\nabla F_{m_k}(\mathbf{x}_k)\|^2+\hat{\gamma}_k\|\nabla F(\hat{\mathbf{x}}_k)-\nabla F(\mathbf{x}_k)\|^2+\hat{\gamma}_k\zeta^2+\frac{\hat{\gamma}_k^2L\sigma^2}{2}\right]\\
&\qquad\le\mathbb{E}\left[-\frac{\hat{\gamma}_k}{8}\|\nabla F(\mathbf{x}_k)\|^2+L^2\hat{\gamma}_k\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|^2+\frac{5\hat{\gamma}_k\zeta^2}{4}+\frac{\hat{\gamma}_k^2L\sigma^2}{2}\right].
\end{aligned}$$

For the third inequality, we used that $\gamma_k\le\frac{1}{2L}$ for all k. For the fifth, we used that $\|\nabla F(\mathbf{x})\|^2\le 2\|\nabla F_m(\mathbf{x})\|^2+2\zeta^2$ for any m and $\mathbf{x}$. Rearranging and summing over k, we

then have

$$\mathbb{E}\left[\sum_{k=1}^{K}\hat{\gamma}_k\|\nabla F(\mathbf{x}_k)\|^2\right] \leq \mathbb{E}\left[8(F(\hat{\mathbf{x}}_1)-F^*)+\sum_{k=1}^{K}\left[8L^2\hat{\gamma}_k\|\mathbf{x}_k-\hat{\mathbf{x}}_k\|^2+10\hat{\gamma}_k\zeta^2+4\hat{\gamma}_k^2L\sigma^2\right]\right].$$

Now, we apply Lemma 3.D.2 to bound the second term on the right hand side:

$$\begin{aligned}&\mathbb{E}\left[\sum_{k=1}^{K}\hat{\gamma}_k\|\nabla F(\mathbf{x}_k)\|^2\right]\\&\leq \mathbb{E}\left[16(F(\hat{\mathbf{x}}_1)-F^*)+\frac{\Delta}{2}+2L\gamma_{\max}\sigma^2\left[M\gamma_{\max}+2\sum_{k=1}^{K}\hat{\gamma}_k\right]+\zeta^2\left[4LM^2\gamma_{\max}^2+11\sum_{k=1}^{K}\hat{\gamma}_k\right]\right].\end{aligned}$$

Therefore, if we choose an output $\tilde{\mathbf{x}}_K$ according to

$$\mathbb{P}(\tilde{\mathbf{x}}_K=\mathbf{x}_k)\propto\hat{\gamma}_k\quad\forall k\in[K],\tag{3.14}$$

then $\mathbb{E}[\|\nabla F(\tilde{\mathbf{x}}_K)\|^2$ is less than this previous expression divided by $\sum_{k=1}^{K}\hat{\gamma}_k$. By exactly the same argument as for Lemma 3.C.2, we can lower bound this sum as:

$$\sum_{k=1}^{K}\hat{\gamma}_k\geq\frac{K\gamma_{\max}}{18}.$$

Therefore, for some constant c (which may change from line to line), we have

$$\begin{aligned}&\mathbb{E}[\|\nabla F(\tilde{\mathbf{x}}_K)\|^2]\\&\leq c\cdot\mathbb{E}\left[\frac{(F(\hat{\mathbf{x}}_1)-F^*)}{K\gamma_{\max}}+\frac{\Delta}{K\gamma_{\max}}+L\gamma_{\max}\sigma^2\left[\frac{M}{K}+1\right]+\zeta^2\left[\frac{LM^2\gamma_{\max}}{K}+1\right]\right]\\&\leq c\cdot\mathbb{E}\left[\frac{(F(\hat{\mathbf{x}}_1)-F^*)}{K\gamma_{\max}}+\frac{\Delta}{K\gamma_{\max}}+L\gamma_{\max}\sigma^2+\zeta^2\right].\end{aligned}$$

Here, we used that $\gamma_{\max}\leq 1/(4ML)$ and $M\leq K$. To conclude, we bound

$$\begin{aligned}&\mathbb{E}[F(\hat{\mathbf{x}}_1)-F^*]\\&=\mathbb{E}\left[F\left[\mathbf{x}_0-\sum_{m=1}^{M}\gamma_{\text{next}(1,m)}\nabla f_m(\mathbf{x}_0;\xi_0^m)\right]-F^*\right]\\&\leq\Delta+\mathbb{E}\left[-\langle\nabla F(\mathbf{x}_0),\sum_{m=1}^{M}\gamma_{\text{next}(1,m)}\nabla F_m(\mathbf{x}_0)\rangle+\frac{L}{2}\left\|\sum_{m=1}^{M}\gamma_{\text{next}(1,m)}\nabla f_m(\mathbf{x}_0;\xi_0^m)\right\|^2\right]\\&\leq\Delta+\mathbb{E}\left[\frac{1}{2L}\|\nabla F(\mathbf{x}_0)\|^2+L\left\|\sum_{m=1}^{M}\gamma_{\text{next}(1,m)}\nabla F_m(\mathbf{x}_0)\right\|^2+\frac{L\sigma^2}{2}\sum_{m=1}^{M}\gamma_{\text{next}(1,m)}^2\right]\\&\leq 2\Delta+\mathbb{E}\left[L\left[2\|\nabla F(\mathbf{x}_0)\|^2+2\zeta^2\right]M\sum_{m=1}^{M}\gamma_{\text{next}(1,m)}^2+\frac{ML\sigma^2\gamma_{\max}^2}{2}\right]\\&\leq 2\Delta+\mathbb{E}\left[LM^2\gamma_{\max}^2\left[4L\Delta+2\zeta^2\right]+\frac{ML\sigma^2\gamma_{\max}^2}{2}\right]\\&\leq 3\Delta+\mathbb{E}\left[\frac{K\gamma_{\max}\zeta^2}{2}+\frac{KL\sigma^2\gamma_{\max}^2}{2}\right].\end{aligned}$$

Plugging this and $\gamma_{\max}$ in above completes the proof. □

Chapter 4

The asynchronous speedup in decentralized optimization

We now turn to studying the impact of delays on the convergence of decentralized methods. While the previous chapter studied computation delays in centralized optimization and showed that asynchrony always benefits the optimizer, through an explicit asynchronous speedup that involves the harmonic mean of computation latencies, this chapter generalizes this result to delayed gossip algorithms, in a stylized framework.

In decentralized optimization, nodes of a communication network each possess a local objective function, and communicate using gossip-based methods in order to minimize the average of these per-node functions. While synchronous algorithms are heavily impacted by a few slow nodes or edges in the graph (the *straggler problem*), their asynchronous counterparts are notoriously harder to parametrize. Indeed, their convergence properties for networks with heterogeneous communication and computation delays have defied analysis so far.

In this chapter, we use the *continuized* framework introduced in Chapter 2 to analyze asynchronous algorithms in networks with delays. Our approach yields a precise characterization of convergence time and of its dependency on heterogeneous delays in the network. Our continuized framework benefits from the best of both continuous and discrete worlds: the algorithms it applies to are based on event-driven updates. They are thus essentially discrete and hence readily implementable. Yet their analysis is essentially in continuous time, relying in part on the theory of delayed ODEs.

Our algorithms moreover achieve an *asynchronous speedup* that is the decentralized analog of that of Chapter 3: the rate of convergence of our delayed decentralized algorithms is controlled by the eigengap of the network graph weighted by *local delays*, instead of the network-wide worst-case delay as in previous analyses. Our methods thus enjoy improved robustness to stragglers.

4.1. Introduction

We consider solving stochastic optimization problems that are distributed amongst n agents (indexed by $\mathcal{V} = [n]$) who can compute stochastic gradients in parallel. This includes classical federated setups, such as distributed and federated learning. Depending on the application, agents have access to either same shared data distribution or a different agent-specific distributions. With *communication cost* being one of the major bottlenecks of parallel optimization algorithms, there are several directions aimed to improve communication efficiency. Amongst the others (such as local update steps [Sti18, WPS+20a] and communication compression [AGL+17, KSJ19]), **decentralization** and **asynchrony** are the two popular techniques for reducing the communication time. Decentralization [KLB+20, LZZ+17] eliminates the dependency on the central server—frequently a major bottleneck in distributed learning—while naturally amplifying privacy guarantees [CEBM22]. Asynchrony [RRWN11, Bau78, TBA86]

shortens the time per computation rounds and allows more updates to be made during the same period of time. It aims to overcome several possible sources of delays: nodes may have *heterogeneous hardware* with different computational throughputs [KMA+19, HLA+21], *network latency* can slow the communication of gradients, and nodes may even just *drop out* [RGPP21]. Moreover, slower "*straggler*" compute nodes can arise in many natural parallel settings, including training ML models using multiple GPUs [CPM+16] or in the cloud; sensitivity to these stragglers poses a serious problem for synchronous algorithms, that depend on the slowest agent. In decentralized synchronous optimization where communication times between pairs of nodes may be heterogeneous, the algorithm can even be further slowed down by *straggling communication links*.

4.1.1. Decentralized and asynchronous setting

More formally, this chapter studies the following optimization problem:

$$\min_{x\in\mathbb{R}^d} \left\{ f(x) = \sum_{v\in\mathcal{V}} f_v(x) \right\}, \tag{4.1}$$

where each individual function $f_v : \mathbb{R}^d \to \mathbb{R}$ for $v \in [n]$ is held by an agent $v\mathcal{V}$, and we consider *asynchronous* and *decentralized* optimization methods that do not rely on a central coordinator. In the case of empirical risk minimization, f_v represents the empirical risk for the local dataset of node v, and f the empirical risk over all datasets. Another important example, that plays the role of a toy problem for both decentralized and/or stochastic optimization is that of **network averaging** [BGPS06], corresponding to $f_v(x) = \|x - c_v\|^2$ where c_v is a vector attached to node v. In this case, the solution of Problem (4.1) reads $\bar{c} = \frac{1}{n}\sum_{v\in\mathcal{V}}^{n} c_v$.

We assume that agents are located at the nodes of a connected, undirected graph $G = (\mathcal{V}, \mathcal{E})$ with node set $\mathcal{V} = [n]$. An agent $v \in \mathcal{V}$ can compute first-order quantities (gradients) related to its local objective function f_v, and can communicate with any adjacent agent in the graph. Our model of asynchrony derives from the popular randomized gossip model of [BGPS06]. In this model, nodes update their local values at random activation times using pairwise communication updates. This asynchronous model makes the idealized assumption of instantaneous communications, and hence does not faithfully represent practical implementations. To alleviate this drawback, several works [AR21, SY18, WYL+18, WLHL15, LCDW16, LZZL18, TSS20, ZY21] introduce communication and computation delays in either pairwise updates, or in asymmetric gossip communications.

However, all these works provide convergence guarantees that either require global synchronization between the nodes, or are implicitly determined by an upper bound on the worst-case delay in the whole graph. Indeed they assume that for some given $k_{\max} > 0$, for all edges $\{v, w\} \in \mathcal{E}$, each communication between agents v and w overlaps with at most $k_{\max}$ other communications in the whole graph. Thus assuming distributed asynchronous operation where individual nodes schedule their interactions based only on local information, the $k_{\max}$ constraint can only be enforced by requiring individual nodes to limit their update frequency to $1/(n\tau_{\max})$, while the resulting algorithms have temporal convergence guarantees that need to be proportional to $\tau_{\max}$. They are thus not robust to *stragglers*, i.e. slow nodes or edges in the graph that induce large $\tau_{\max}$. Moreover, all these works assume that agents v or graph edges $\{v, w\}$ are activated for agent interaction sequentially in an *i.i.d.* manner, which can only be unforced with a central coordinator, and can lead to deadlocks.

4.1.2. Graph-dependent asynchronous speedup

We now describe the notion of asynchronous speedup explained in Section 1.3 for centralized and decentralized settings. To understand the scope for improvement over methods that rely on such worst-case delays or over synchronous algorithms, recall that for synchronous algorithms with updates performed every $\tau_{\max}$ seconds, for L-smooth and σ-strongly con-

vex functions f_v, the time required to reach precision $\varepsilon > 0$ for $\frac{1}{n}\sum f_v$ is lower-bounded by [SBB+17]:

$$\Omega\left(\tau_{\max}\mathrm{Diam}(G)\sqrt{\kappa}\ln(\varepsilon^{-1})\right)\,, \tag{4.2}$$

where $\kappa = L/\sigma$ is the condition number of the functions f_v and $\mathrm{Diam}(G)$ is the diameter of graph G.

In this chapter we seek better dependency on individual delays in the network, as we did in the previous chapter in the centralized setting. Specifically we consider the following.

Assumption 4.1.1 (**Heterogeneous delays**). *There exist $\tau_{\{v,w\}}$ for $\{v,w\} \in \mathcal{E}$ and τ_v^{comp} for $v \in \mathcal{V}$ such that communications between two neighboring agents v and w in the graph take time at most $\tau_{\{v,w\}}$, and a computation at node v takes time at most τ_v^{comp}.*

Under such heterogeneous delay assumptions, *how robust to stragglers can decentralized algorithms be?* One can adapt the proof of [SBB+17] to Assumption 4.1.1 to establish the generalized form of lower bound (4.2):

$$\Omega\left(D(\tau)\sqrt{\kappa}\ln(\varepsilon^{-1})\right)\,, \tag{4.3}$$

where $D(\tau) = \sup_{(v,w)\in\mathcal{V}^2}\mathrm{dist}(v,w)$ for:

$$\mathrm{dist}(v,w) = \inf_{\substack{(v=v_0,\dots,v_p=w),\\ \forall 1\leq k\leq p,\,(v_k,v_{k+1})\in\mathcal{E}}} \tau_v^{\mathrm{comp}} + \tau_w^{\mathrm{comp}} + \sum_{k=0}^{p-1}\tau_{v_kv_{k+1}}\,.$$

Here $\mathrm{dist}(v,w)$ is the time distance between nodes v and w, and $D(\tau)$ is the *diameter* of graph G for this distance. $D(\tau)$ is the generalization of $\tau_{\max}\mathrm{Diam}(G)$ to the heterogeneous-delay setting. This lower bound suggests that robustness to stragglers is possible: indeed if a fraction of the nodes or edges is too slow (large delay $\tau_{\{v,w\}}$), this may not even impact this lower bound, since the shortest path between two nodes may always take another route.

We aim at building *decentralized* algorithms with performance guarantees that enjoy such robustness to individual delay bounds. However, since we focus on fully decentralized algorithms, our performance guarantees will not be expressed in terms of some diameter $D(\tau)$ as in (4.3) but instead in terms of some spectral characteristics of the graph at hand[1]. Specifically, let us recall the notion of Graph Laplacian.

Definition 4.1.1 (**Graph Laplacian**). *Let $\nu = (\nu_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$ be a set of non-negative real numbers. The* Laplacian *of the graph G weighted by the $\nu_{\{v,w\}}$'s is the symmetric matrix $\Delta_G(\nu)$ with (v,w) entry equal to $-\nu_{\{v,w\}}$ if $\{v,w\}\in\mathcal{E}$, $\sum_{u\sim v}\nu_{\{u,v\}}$ if $w = v$, and 0 otherwise. In the sequel $\nu_{\{v,w\}}$ always refers to the weights of the Laplacian, and $\lambda_2(\Delta_G(\nu))$ denotes this Laplacian's second smallest eigenvalue.*

We thus seek performance guarantees similar to (4.3) with in place of $D(\tau)$ the term $\lambda_2(\Delta_G(\nu))^{-1}$ for some parameters $\nu_{\{v,w\}}$ that depend on delay characteristics local to edge $\{v,w\}$. As a consequence, we highlight the fact that due to this local dependency on the delays (that no existing work considers), there will always exist graphs and topologies that keep our rate of convergence constant, while making existing approaches (that all depend on worst-case delays) fail to converge in a reasonable amount of time.

4.1.3. Related works specific to this chapter

Decentralized optimization and asynchrony. Previous decentralized asynchronous works are restricted to a given communication protocol and static topologies under an *i.i.d.* sampling of the nodes or edges that become active [AR21, LHLL15, BRW+23, NSD+21], no

[1]Note that similar spectral characteristics (albeit based on a single worst-case delay parameter $\tau_{\max}$) appear in [AR21, SY18, WYL+18, WLHL15, LCDW16, SBB+17].

communication delays [LHLL15, BRW+23, NSD+21], or their analyses rely on an upper-bound on the maximal computation and communication delays [AR21, LZZL18, BRW+23, LHZQ20, LYW+22, NSD+21, ZY21, WLMJ23]. This latter point is exactly the goal of this chapter: how can we relax the worst case delay dependency and capture quantitatively delay heterogeneity in the graph and relate it to a *wall-clock asynchronous speedup*? Similarly to the asynchronous speedup of Asynchronous SGD described just above, we will quantify the asynchronous speedup as a graph dependent quantity that takes into account pairwise communication delays and their heterogeneity.

We will here deal with asynchrony and delays from a different viewpoint than the references cited above, where delays are introduced in the analysis as discrete-time delays that are uniformly bounded by some given quantity, while our analysis is inspired by time-delayed ODE systems [Nic01]. Consequently, the assumptions related to delays and asynchrony (such as Assumption 4.1.1) do not need to be translated into discrete-time ones, as in the above references. Finally, we believe our continuous time framework to be particularly adequate for the study and design of asynchronous algorithms, in the decentralized setting as in this chapter, but also in centralized settings where it may remove the need to introduce a discrete ordering of events and thus avoid difficulties that lead to unrealistic assumptions, such as the *after/before-read* approaches [LPLJ18].

Finally, [WSY+19] study how sparsifying the communication graph can lead to faster decentralized algorithm. Their approach is different from ours in Section 4.6: they do not consider asynchronous algorithms with physical constraints (delays and capacity), but synchronous algorithms where sequentially matchings are built in the graph. Yet, we observe similar phenomenon as theirs in Section 4.6.

Network time protocol (NTP). We here implicitly assume that nodes of the graph are aware of a global absolute time (continuous-time), which is a feature of the *continuized* framework, presented in a subsequent subsection. This is very different from knowing a global (discrete) iteration counter that tracks the number of updates. The latter is impossible in a decentralized framework, while the former can be achieved fairly easily, up to some synchronization errors. In its standard version, the Network Time Protocol (*NTP*) ensures that machines connected to internet have access to the global time with a precision of the millisecond. In our case, in order to achieve better precision, more refined versions of NTPs can be used, such as [GLY+18], that yields a precision of the order of the nanosecond (10^{-9} seconds). Hence, up to negligible errors, agents indeed share a global continuous-time clock.

4.1.4. Outline of the chapter

(i) We first consider the *network averaging problem*, for which we introduce *Delayed Randomized Gossip* in Section 4.2. Building on the continuized framework introduced in Chapter 2, we analyze Delayed Randomized Gossip in the continuized framework, that allows a continuous-time analysis of an algorithm even though the latter is based on discrete, hence practically implementable operations. Our analysis leads to explicit stability conditions that have the appealing property of being *local*, i.e. they require each agent to tune its algorithm parameters to delay bounds in its graph neighborhood.

They ensure a linear rate of convergence determined by $\lambda_2\big(\Delta_G(\nu)\big)$, for weights of order $\nu = 1/(\sum_{\{v',w'\}\sim\{v,w\}} \tau_{\{v',w'\}})$[2]. This dependency of weights in the Laplacian on local delay bounds is what we call the *asynchronous speedup*, since it implies a scaling that is no longer proportional to $\tau_{\max}$.

(ii) Using an augmented graph approach, we propose algorithms that generalize Delayed Randomized Gossip to solve the decentralized optimization problem in Section 4.4. Under strong convexity and smoothness assumptions on the local functions f_v, we obtain local stability conditions yielding an asynchronous speedup for this more general setup.

[2]We write $\{v,w\} \sim \{v',w'\}$ and say that two edges $\{v,w\}$, $\{v',w'\}$ are neighbors if they share at least one node.

(iii) We further generalize our setup with the introduction of *local capacity constraints* in Section 4.5, in order to take into account the fact that nodes or edges cannot handle an unlimited number of operations in parallel. To that end, we introduce *truncated Poisson point processes* in the continuized framework for the analysis.

(iv) The theoretical guarantees for our algorithms in Sections 4.2, 4.4 and 4.5 are all based on general guarantees for so-called delayed coordinate gradient descent in the continuized framework, that we present and establish in Section 4.3. These results may be of independent interest beyond our current focus on decentralized optimization.

(v) Finally, we identify from our stability conditions and convergence guarantees a phenomenon reminiscent of *Braess*'s paradox (Section 4.6): deleting some carefully chosen edges can lead to faster convergence. This in turn suggests rules for sparsifying communication networks in distributed optimization.

This chapter is thus organized in a linear way, where difficulties are added one by one for pedagogical sake. We start by adding delays in communications (4.2) before providing the necessary tools for the analysis (4.3). We then build on this to progressively add local functions (4.4) and local capacity constraints (4.5).

4.2. Delayed Randomized Gossip for Network Averaging

Focusing in this section on the Network Averaging Problem, we introduce Delayed Randomized Gossip and state its convergence guarantees. We first begin with reminders on randomized gossip [BGPS06].

4.2.1. Randomized gossip

Each agent $v \in \mathcal{V}$ is assigned a real vector $x_0(v) \in \mathbb{R}^d$. The goal of the averaging (or gossip) problem is to design an iterative procedure allowing each agent in the network to estimate the average $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_0(v)$ using only local communications, *i.e.*, communications between adjacent agents in the network.

In randomized gossip [BGPS06], time t is indexed continuously by $\mathbb{R}^+$. A Poisson point process [Kle14] (abbreviated as *P.p.p.* in the sequel) $\mathcal{P} = \{T_k\}_{k\geq 1}$ of intensity $I > 0$ on $\mathbb{R}^+$ is generated: $T_0 = 0$ and $(T_{k+1} - T_k)_{k\geq 0}$ are *i.i.d.* exponential random variables of mean $1/I$. For positive intensities $(p_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$ such that $\sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}} = I$, for every $k \geq 0$, at T_k an edge $\{v_k, w_k\}$ is *activated* with probability $p_{\{v_k,w_k\}}/I$, upon which adjacent nodes v_k and w_k communicate and perform a pairwise update. The *P.p.p.* assumption implies that edges are activated independently of one another and from the past: the activation times of edge $\{v, w\}$ form a *P.p.p.* of intensity $p_{\{v,w\}}$. To solve the gossip problem, [BGPS06] proposed the following strategy: each agent $v \in \mathcal{V}$ keeps a local estimate $x_t(v)$ of the average and, upon activation of edge $\{v_k, w_k\}$ at time $T_k \in \mathbb{R}^+$, the activated nodes v_k, w_k average their current estimates:

$$x_{T_k}(v_k),\, x_{T_k}(w_k) \longleftarrow \frac{x_{T_k-}(v_k) + x_{T_k-}(w_k)}{2}\,. \tag{4.4}$$

Writing $f(x) = \sum_{\{v,w\}\in\mathcal{E}} \frac{p_{\{v,w\}}}{I} f_{\{v,w\}}(x)$, for

$$f_{\{v,w\}}(x) = \frac{1}{2}\|x(v) - x(w)\|^2 \quad \text{and} \quad x = (x(v))_{v\in\mathcal{V}}\,, \tag{4.5}$$

we observed in Chapter 2 that local averages (4.4) correspond to stochastic gradient steps on function f:

$$x_{T_k} \longleftarrow x_{T_k-} - \frac{K_{\{v_k,w_k\}}}{p_{\{v_k,w_k\}}} \nabla f_{\{v_k,w_k\}}(x_{T_k-})\,, \tag{4.6}$$

for step sizes $K_{\{v_k,w_k\}} = \frac{p_{\{v_k,w_k\}}}{2}$.

These updates can also be derived from coordinate gradient descent steps. Let $A \in \mathbb{R}^{\mathcal{V}\times\mathcal{E}}$ be such that for all $\{v,w\} \in \mathcal{E}$, $Ae_{\{v,w\}} = \mu_{\{v,w\}}(e_v - e_w)$ for arbitrary $\mu_{\{v,w\}} \in \mathbb{R}$, where $(e_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$ and $(e_v)_{v\in\mathcal{V}}$ are the canonical bases of $\mathbb{R}^E$ and $\mathbb{R}^V$. Then, let $g(\lambda) = \frac{1}{2}\|A\lambda\|^2$ for $\lambda \in \mathbb{R}^{\mathcal{E}\times d}$, so that the coordinate gradient $\nabla_{\{v,w\}}g(\lambda) \in \mathbb{R}^d$ writes $\nabla_{\{v,w\}}g(\lambda) = \mu_{\{v,w\}}((A\lambda)_v - (A\lambda)_w)$. Depending on the context, we will either consider $\nabla_{\{v,w\}}g(\lambda)$ as a vector in $\mathbb{R}^d$, or by an abuse of notation, write $\nabla_{\{v,w\}}g(\lambda) \in \mathbb{R}^{\mathcal{D}\times d}$ the matrix vector whose rows are all null, except the row corresponding to edge $\{v,w\}$, that is equal to $\mu_{\{v,w\}}((A\lambda)_v - (A\lambda)_w)$. This amounts to write $\nabla_{\{v,w\}}g(\lambda)$ instead of $e_{\{v,w\}}\nabla_{\{v,w\}}g(\lambda)^\top$. Thus, provided that for some $\lambda_{T_k-} \in \mathbb{R}^{\mathcal{E}\times d}$, $x_{T_k-} - \bar{x} = A\lambda_{T_k-}$, the local averaging defined in Equation (4.4) is equivalent to $x_{T_k} - \bar{x} = A\lambda_{T_k}$, where:

$$\lambda_{T_k} = \lambda_{T_k-} - \frac{K_{\{v_k,w_k\}}}{p_{\{v_k,w_k\}}\mu^2_{\{v_k,w_k\}}}\nabla_{\{v_k,w_k\}}g(\lambda_{T_k-})\,, \tag{4.7}$$

for $K_{\{v_k,w_k\}} = \frac{p_{\{v_k,w_k\}}}{2}$. Hence, the gossip algorithm of [BGPS06] can be viewed as a simple block-coordinate gradient descent on variables $\lambda \in \mathbb{R}^{\mathcal{E}\times d}$ indexed by the edges of the graph instead of the nodes.

Yet, this continuous-time model with *P.p.p.* activations implicitly assumes instantaneous communications, or some form of waiting – this was the case also in Chapter 2. Indeed, the gradient is computed on the current value of the parameter, which is x_{T_k-}. In the presence of (heterogeneous) communication delays (Assumption 4.1.1), a more realistic update uses the parameter x_{S_k} at a previous time $S_k < T_k$, to account for the time it takes to compute and communicate the gradient. In this case, the updates write as

$$x_{T_k} \longleftarrow x_{T_k-} - \frac{K_{\{v_k,w_k\}}}{p_{\{v_k,w_k\}}}\nabla f_{\{v_k,w_k\}}(x_{S_k})\,. \tag{4.8}$$

Equivalently, from the point of view of node v_k:

$$x_{T_k}(v_k) \leftarrow x_{T_k-}(v_k) - \frac{K_{\{v_k,w_k\}}}{p_{\{v_k,w_k\}}}\big(x_{S_k}(v_k) - x_{S_k}(w_k)\big)\,.$$

4.2.2. Delays in the continuized framework

Our approach uses the **continuized framework** (Chapter 2), which amounts to consider continuous-time evolution of key quantities, with discrete jumps at the instants of Poisson point processes. This gives the best of both continuous (for the analysis and assumptions) and discrete (for the implementation) worlds. From now on and for the rest of the chapter, we assume that Assumption 4.1.1 holds.

While in Section 2.7 in Chapter 2 local *P.p.p.* were all of the same intensity, we here relax homogeneity. Edges $\{v,w\} \in \mathcal{E}$ locally generate independent *P.p.p.* $\mathcal{P}_{\{v,w\}}$ of intensity $p_{\{v,w\}} > 0$ (random activation times, with *i.i.d.* intervals, exponentially distributed with mean $1/p_{\{v,w\}}$). As mentioned previously, $\mathcal{P} = \bigcup_{\{v,w\}\in\mathcal{E}}\mathcal{P}_{\{v,w\}}$ is a *P.p.p.* of intensity $I = \sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}}$, and noting $\mathcal{P} = \{T_1 < T_2 < \dots\}$, at each clock ticking T_k, $k \geq 1$, an edge $\{v_k, w_k\}$ is chosen with probability $p_{\{v_k,w_k\}}/I$. This time T_k corresponds to a communication update between nodes v_k and w_k started at time $T_k - \tau_{\{v_k,w_k\}}$ [3]. Assumption 4.1.1 ensures that the communication started at time $T_k - \tau_{\{v,w\}}$ takes some time $\tau^{(k)} \leq \tau_{\{v_k,w_k\}}$ and is thus completed before time T_k so that the update at time T_k is indeed implementable.

[3] Standard properties of P.p.p. guarantee that the sequence of points of $\mathcal{P}_{\{v,w\}}$ translated by $\tau_{\{v,w\}}$ is a P.p.p. with the same distribution.

Algorithm 4.1: Delayed randomized gossip, edge $\{v, w\}$

Step size $K_{\{v,w\}} > 0$ and intensity $p_{\{v,w\}} > 0$
Initialization $T_1\{v, w\} \sim \text{Exp}(p_{\{v,w\}})$
for $\ell = 1, 2, \ldots$ **do**
  $T_{\ell+1}\{v, w\} = T_\ell\{v, w\} + \text{Exp}(p_{\{v,w\}})$.
end for
for $\ell = 1, 2, \ldots$ **do**
  At time $T_\ell\{v, w\} - \tau_{\{v,w\}}$ for, v sends $\hat{x}_v = x_{T_\ell\{v,w\}-\tau_{\{v,w\}}}(v)$ to w and w sends $\hat{x}_w = x_{T_\ell\{v,w\}-\tau_{\{v,w\}}}(w)$ to v.
  At time $T_\ell\{v, w\}$,

$$\begin{aligned} x_{T_\ell\{v,w\}}(v) &\leftarrow x_{T_\ell\{v,w\}-}(v) - \frac{K_{\{v,w\}}}{p_{\{v,w\}}}\big(\hat{x}_v - \hat{x}_w\big)\,, \\ x_{T_\ell\{v,w\}}(w) &\leftarrow x_{T_\ell\{v,w\}-}(w) - \frac{K_{\{v,w\}}}{p_{\{v,w\}}}\big(\hat{x}_w - \hat{x}_v\big)\,, \end{aligned} \tag{4.10}$$

end for

Consequently, the sequence $(x_t)_t$ generated by Algorithm 4.1 writes as:

$$\begin{cases} x_{T_k}(v) = x_{T_k-}(v) \quad \text{if} \quad v \notin \{v_k, w_k\}\,, \\ x_{T_k}(v_k) \leftarrow x_{T_k-}(v_k) - \dfrac{K_{\{v_k,w_k\}}}{p_{\{v_k,w_k\}}}\big(x_{T_k-\tau_{\{v_k,w_k\}}}(v_k) - x_{T_k-\tau_{\{v_k,w_k\}}}(w_k)\big), \\ x_{T_k}(w_k) \leftarrow x_{T_k-}(w_k) - \dfrac{K_{\{v_k,w_k\}}}{p_{\{v_k,w_k\}}}\big(x_{T_k-\tau_{\{v_k,w_k\}}}(w_k) - x_{T_k-\tau_{\{v_k,w_k\}}}(v_k)\big). \end{cases}$$

Algorithm 4.1 is the pseudo-code for *Delayed Randomized Gossip*, from the viewpoint of two adjacent nodes v and w. The times $T_\ell\{v, w\}$ for $\ell \geq 1$ denote the activation times of edge $\{v, w\}$. They follow a P.p.p. of intensity $p_{\{v,w\}}$, and are sequentially determined by adjacent nodes v and w. $\text{Exp}(p)$ is an exponential random variable of parameter p. In Algorithm 4.1, Delayed Randomized Gossip is presented from the local viewpoint of edges $\{v, w\} \in \mathcal{E}$ ($T_\ell\{v, w\}$ is the ℓ^{th} activation of edge $\{v, w\}$), while the equations just above are a global description of the algorithm (T_k is the k^{th} edges activation in the graph).

Formally, this decentralized and asynchronous algorithm corresponds to a jump process solution of a *delayed stochastic differential equation*. Defining $N(\mathrm{d}t, \{v, w\})$ as the *Poisson* measure on $\mathbb{R}^+ \times E$ of intensity $I\mathrm{d}t \otimes \mathcal{U}_p$ where $\mathcal{U}_p$ is the probability distribution on E proportional to $(p_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$ ($\mathcal{U}_p(\{v, w\}) = p_{\{v,w\}}/I$), we have:

$$\mathrm{d}x_t = -\int_{\mathbb{R}^+\times E} \frac{K_{\{v,w\}}}{p_{\{v,w\}}} \nabla f_{\{v,w\}}(x_{t-\tau_{\{v,w\}}}) \mathrm{d}N(t, \{v, w\})\,. \tag{4.9}$$

Next section presents convergence guarantees for iterates generated by delayed randomized gossip.

4.2.3. Convergence guarantees

We begin by recalling the key quantities introduced. *(i)* The constraints inherent to the problem are the communication delays, upper-bounded by constants $\tau_{\{v,w\}}$. *(ii)* Parameters of the algorithm are: step sizes $K_{\{v,w\}} > 0$ and intensities $p_{\{v,w\}}$ of the local P.p.p. that

trigger communications between adjacent nodes v and w. For arbitrary intensities $p_{\{v,w\}}$ and delay bounds $\tau_{\{v,w\}}$, we shall provide local conditions on the step sizes $K_{\{v,w\}}$ that guarantee stability and convergence guarantees. This is to be contrasted with the situation –discussed in Section 4.5– where in addition there are capacity constraints, for which additional conditions on the intensities $p_{\{v,w\}}$ are needed to prove convergence.

Theorem 4.1 (**Delayed Randomized Gossip**). *Assume that for all $\{v,w\} \in \mathcal{E}$, we have*[4]*:*

$$K_{\{v,w\}} \leq \frac{p_{\{v,w\}}}{1 + \sum_{\{v',w'\} \sim \{v,w\}} p_{\{v',w'\}} \big(\tau_{\{v,w\}} + e\tau_{\{v',w'\}}\big)} \,. \tag{4.11}$$

Let $\nu_{\{v,w\}} \equiv K_{\{v,w\}}$, $\{v,w\} \in \mathcal{E}$, and $\tau_{\max} = \max_{\{v,w\} \in \mathcal{E}} \tau_{\{v,w\}}$. Let $\gamma > 0$ be such that:

$$\gamma \leq \min\left(\frac{\lambda_2\big(\Delta_G(\nu)\big)}{2} \,,\, \frac{1}{\tau_{\max}} \right) .$$

For any $T \geq 0$, for $(x_t)_{t \geq 0}$ generated with delayed randomized gossip (Algorithm 4.1) or equivalently by the delayed SDE in Equation (4.9), we have:

$$\frac{\int_0^T e^{\gamma t} \mathbb{E}\left[\|x_t - \bar{x}\|^2 \right] \mathrm{d}t}{\int_0^T e^{\gamma t} \|x_0 - \bar{x}\|^2 \mathrm{d}t} \leq e^{-\frac{\gamma T}{2}} \frac{1 + \frac{\tau_{\max}}{T}}{1 - \gamma \tau_{\max}} \,. \tag{4.12}$$

The proof is deferred to a subsequent subsection. Using Jensen inequality then yields the following corollary: a weighted average of the iterates is decreasing linearly with time. This is a countinuous-time counterpart of the weighted average considered in most decentralized optimization algorithms (strongly convex case in [KLB+20], e.g.). Note that this integral is in fact discrete and can be expressed as a sum, since $(x_s)_s$ is a jump process. Finally, to converse this result in discrete time (number of pairwise communications), we just need to notice that the number of communications that happened before time $T \geq 0$ is equal in mean to $T \sum_{\{v,w\} \in \mathcal{E}} p_{\{v,w\}}$, and concentrates around this value for T large.

Corollary 4.2.1. *Under the same assumptions as Theorem 4.1, for $(x_t)_{t \geq 0}$ generated with delayed randomized gossip, define $(\tilde{x}_t)_{t \geq 0}$ as the exponentially weighted averaging along the trajectory of (x_t):*

$$\tilde{x}_t = \gamma \frac{\int_0^t e^{\gamma s} x_s \mathrm{d}s}{e^{\gamma t} - 1} \,.$$

Then, for all $T \geq 0$,

$$\mathbb{E}\left[\|\tilde{x}_T - \bar{x}\|^2 \right] \leq e^{-\frac{\gamma T}{2}} \|x_0 - \bar{x}\|^2 \frac{1 + \frac{\tau_{\max}}{T}}{1 - \gamma \tau_{\max}} \,.$$

An essential aspect of Theorem 4.1 lies in the explicit sufficient conditions for convergence it establishes for our proposed schemes, and on how they only rely on (upper bounds on) individual delays. We now discuss the **asynchronous speedup** obtained by fine-tuning algorithm parameters according to delays.

For many graphs of interest such as grids, hypergrids, trees... in the large network limit $n \to \infty$ one has $\lambda_2(\Delta_G(\nu)) \to 0$[5] and so $\min(\lambda_2(\Delta_G), 1/\tau_{\max}) = \lambda_2(\Delta_G)$ should hold. More precisely, let Λ_n be the smallest non-null eigenvalue of the Laplacian with weights equal to $1/\mathrm{degree}_{\{v,w\}}$ (the degree of an edge). Then, if for instance $p_{\{v,w\}} = 1/\tau_{\{v,w\}}$, as long as $\Lambda_n = \mathcal{O}(\frac{\tau_{\min}}{d_{\max}\tau_{\max}})$ (for the line/cyclic graphs, $\Lambda_n = \mathcal{O}(1/n^2)$, for the $D-$dimensional grid

[4] Note that $\{v,w\} \sim \{v,w\}$; constant e is $\exp(1)$.

[5] Networks for which this fails are known as *expanders*.

$\Lambda_n = \mathcal{O}(1/n^{2/D})$, and this condition will hold in most cases) where $d_{\max}$ is the max degree in the graph, we have $\gamma = \frac{\lambda_2\left(\Delta_G(\nu)\right)}{2}$. However, for small graphs that do not verify the condition $\Lambda_n = \mathcal{O}(\frac{\tau_{\min}}{\tau_{\max}})$, the linear rate of convergence turns into $\frac{1}{\tau_{\max}}$. Noting that synchronous gossip has a linear rate of convergence of $\frac{\Lambda_n}{\tau_{\max}}$, we still notice an improvement of magnitude Λ_n for such graphs, and our decentralized approach behaves as if it were centralized. The *asynchronous speedup* consists in having a rate of convergence as the eigengap of the Laplacian of the graph weighted by local communication constraints: this is thus the case here, with the term $\lambda_2(\Delta_G(K))$, where each $K_{\{v,w\}}$ is impacted only by local quantities.

As mentioned in the introduction, this quantity should be understood as the analogue in decentralized optimization of the squared diameter of the graph (using time distances) in (4.3) in centrally coordinated algorithms and as expected, gossip algorithms are affected by spectral properties of the graph. In Theorem 4.1, these properties reflect delay heterogeneity across the graph: here, $\lambda_2(\Delta_G(K))^{-1}$ the mixing time of a random walk on the graph where jumping from node v to w takes a time $\tilde{\tau}_{\{v,w\}} = K^{-1}_{\{v,w\}}$. In contrast, previous analyses (of synchronous or asynchronous algorithms) involve the mixing time of a random walk with times between jumps set to a quantity that is linearly dependent on $\tau_{\max}$. We refer to this ratio as the *asynchronous speedup*.

Equation (4.11) suggests a scaling of $p_{\{v,w\}} \approx 1/\tau_{\{v,w\}}$, giving local weights $K_{\{v,w\}}$ of order $1/(\text{degree}_{\{v,w\}}\tau_{\{v,w\}})$ where $\text{degree}_{\{v,w\}}$ is the degree of edge $\{v,w\}$ in the edge-edge graph. On the other hand, synchronous algorithms are slowed down by the slowest node: the equivalent term would be of order $\lambda_2(\Delta_G(1/(\text{degree}_{\{v,w\}}\tau_{\max}))$. Indeed, for a *gossip matrix* $W \in \mathbb{R}^{V\times V}$ (W is a symmetric and stochastic matrix), the equivalent factor in synchronous gossip [DKM+10] is $\lambda_2(\Delta_G(W_{\{v,w\}}\tau_{\max}))$, and $W_{\{v,w\}}$ is usually set as $1/\text{degree}_{\{v,w\}}$.

4.2.4. A delayed ODE for mean values in gossip

Before proving Theorem 4.1, we provide some intuition for its conditions and the resulting convergence rate. We do this by studying the means of the iterates, that verify a delayed linear ordinary differential equation, easier to study than the process itself, for which we provide stability conditions. Denoting $y_t = \mathbb{E}[x_t] \in \mathbb{R}^{n\times d}$, for $t \geq 0$, where $(x_t)_{t\geq 0}$ is generated using delayed randomized gossip updates (4.8), we have:

$$\frac{\mathrm{d}y_t}{\mathrm{d}t} = -\sum_{\{v,w\}\in\mathcal{E}} K_{\{v,w\}}\nabla f_{\{v,w\}}(y_{t-\tau_{\{v,w\}}})\,, \tag{4.13}$$

where $f_{\{v,w\}}$ is defined in Equation (4.5). Indeed, for any $t \geq 0$ and $\mathrm{d}t > 0$,

$$\begin{aligned}\mathbb{E}[x_{t+\mathrm{d}t}|x_t] - x_t &= -x_t + (1 - I\mathrm{d}t)x_t + o(\mathrm{d}t)\\ &\quad + \mathrm{d}t\sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}}\Big(x_t - \frac{K_{\{v,w\}}}{p_{\{v,w\}}}\nabla f_{\{v,w\}}(x_{t-\tau_{\{v,w\}}})\Big)\\ &= -\mathrm{d}t\sum_{\{v,w\}\in\mathcal{E}} K_{\{v,w\}}\nabla f_{\{v,w\}}(x_{t-\tau_{\{v,w\}}}) + o(\mathrm{d}t)\,.\end{aligned}$$

Taking the mean, using the linearity of $\nabla f_{\{v,w\}}$, dividing by $\mathrm{d}t$ and making $\mathrm{d}t \to 0$ leads to the delayed ODE verified by $y_t = \mathbb{E}[x_t]$. Such delay-differential ODEs are classical [Nic01] yet their stability properties are notoriously hard to characterize. This is typically attacked by means of *Lyapunov-Krasovskii* functionals or *Lyapunov-Razumikhin* functions [GL09]. Alternatively, sufficient conditions for convergence and stability guarantees on (y_t) can be obtained, under specific conditions, by enforcing stability of the original system after *linearizing* it with respect to delays [Mas02]. Linearizing in the sense of [Mas02] means making

the approximation $y_{t-\tau_{\{v,w\}}} = y_t - \tau_{\{v,w\}} \frac{\mathrm{d}y_t}{\mathrm{d}t}$. Under this approximation, we have:

$$\frac{\mathrm{d}y_t}{\mathrm{d}t} = -\sum_{\{v,w\}\in\mathcal{E}} K_{\{v,w\}} \left(\nabla f_{\{v,w\}}(y_t) - \tau_{\{v,w\}} \nabla f_{\{v,w\}}(\frac{\mathrm{d}y_t}{\mathrm{d}t}) \right).$$

For any weights $\nu_{\{v,w\}}$ and vector z, $\sum_{\{v,w\}} \nu_{\{v,w\}} \nabla f_{\{v,w\}}(z) = \Delta_G(\nu) z$. Thus the delay-linearized ODE reads

$$(\mathrm{Id} - \Delta_G(\{K_{\{v,w\}}\tau_{\{v,w\}}\})) \frac{\mathrm{d}y_t}{\mathrm{d}t} = -\Delta_G(\{K_{\{v,w\}}\}) y_t \,. \tag{4.14}$$

This delay-linearized ODE (4.14) provides intuition on the behavior of $\mathbb{E}[x_t]$. Indeed, (4.14) is stable provided that $\rho(\Delta_G(\{\tau_{\{v,w\}}K_{\{v,w\}}\})) < 1$, in which case it has a linear rate of convergence of order $\lambda_2(\Delta_G(\{K_{\{v,w\}}\})$.

Even though this stability condition and the rate of convergence are only heuristics, since (4.14) is obtained through an approximation of the delayed ODE verified by $\mathbb{E}[x_t]$ (4.13), this stability condition for the delay-linearized system implies stability of the original delayed system under assumptions on the matrices and delays involved[Mas02], that hold in our case, leading to the following

Proposition 4.2.1. *Assume that the spectral radius of the weighted Laplacian $\Delta_G(\{\tau_{\{v,w\}}K_{\{v,w\}}\})$ verifies $\rho(\Delta_G(\{\tau_{\{v,w\}}K_{\{v,w\}}\})) < 1$. Then the delayed* ODE (4.13) *is stable.*

Consequently, the stability conditions (necessary conditions on step sizes $K_{\{v,w\}}$ in Equation (4.11)) obtained in Theorem 4.1 are very natural. Indeed, a simple way to enforce $\rho(\Delta_G(\{\tau_{\{v,w\}}K_{\{v,w\}}\}) < 1$ based on local conditions consists in imposing $\sum_j \tau_{\{v,w\}}K_{\{v,w\}} < 1$ for all v. This is a weaker condition than the one stated in Theorem 4.1, but it only gives stability of the means. Furthermore, the rate of convergence of delayed randomized gossip in Theorem 4.1, that takes the form of the eigengap of a weighted graph Laplacian, is also that of any solution of the delay-linearized ODE (4.14).

Proof. For $A \in \mathbb{R}^{\mathcal{V}\times\mathcal{E}}$ as defined in Section 4.2.1 for non-null weights $\mu_{\{v,w\}}$, define the following delayed ODE:

$$\frac{\mathrm{d}\lambda_t}{\mathrm{d}t} = -\sum_{\{v,w\}\in\mathcal{E}} \frac{K_{\{v,w\}}}{\mu^2_{\{v,w\}}} e_{\{v,w\}}^\top A^\top A \lambda_{t-\tau_{\{v,w\}}} \,. \tag{4.15}$$

For (y_t) solution of (4.13), if there exists λ_0 such that $A\lambda_0 = y_0$, then $y_t = A\lambda_t$ for all t, where λ_t is solution of (4.15) initialized at the value λ_0. Then, since $AA^\top$ is the Laplacian of graph G with weights $\mu^2_{\{v,w\}} > 0$, A is of rank $n-1$. For all λ, $A\lambda$ is in the orthogonal of $\mathbb{R}\mathbb{1}$ ($\mathbb{1} \in \mathbb{R}^V$ is the vector with all entries equal to 1), so that $\mathrm{Im}(A)$ is exactly the orthogonal of $\mathbb{R}\mathbb{1}$. Finally, since for (y_t) a solution of (4.13), $y_t - (\mathbb{1}^\top y_0)\mathbb{1}$ is also solution of (4.13) and takes values in the orthogonal of $\mathbb{R}\mathbb{1}$, it is sufficient to prove stability of (4.15).

To that end, we use Theorem 1 of [Mas02]. For $z \in \mathbb{R}^{\mathcal{E}}$, let $D(z) \in \mathbb{R}^{\mathcal{E}\times\mathcal{E}}$ be the diagonal matrix with diagonal equal to z. Let $M = D(\frac{K}{\mu^2})A^\top A$. Then, the delayed ODE (4.15) writes as:

$$\frac{\mathrm{d}\lambda_t\{v,w\}}{\mathrm{d}t} = -\sum_{\{v',w'\}\in\mathcal{E}} M_{\{v,w\},\{v',w'\}} \lambda_{t-\tau_{\{v,w\}}}(\{v',w'\}) \quad , \{v,w\} \in \mathcal{E} \,,$$

an ODE that takes the same form as Equation (7) in [Mas02], for $D^{\leftarrow}_{\{v,w\}} = \tau_{\{v,w\}}$, $D^{\rightarrow}_{\{v,w\}} = 0$ and $D_{\{v,w\}} = \tau_{\{v,w\}}$, $R = \mathcal{E}$ and with our matrix M. In order to ensure that M is symmetric and positive semi-definite, we take $\mu^2_{\{v,w\}} = K_{\{v,w\}}$, to have $M = A^\top A$. The assumptions of Theorem 1 of [Mas02] are verified, so that the delayed ODE (4.15) is stable if $\rho(D(\tau)M) < 1$. We then write $\rho(D(\tau)M) = \rho(D(\sqrt{\tau})A^\top A D(\sqrt{\tau}) = \rho(AD(\sqrt{\tau})(AD(\sqrt{\tau}))^\top)$, and notice that

$AD(\sqrt{\tau})(AD(\sqrt{\tau}))^\top$ is the Laplacian of graph G with weights $\mu^2_{\{v,w\}}\tau_{\{v,w\}} = K_{\{v,w\}}\tau_{\{v,w\}}$, concluding the proof. □

4.2.5. Proof of Theorem 4.1

In the proof, we use the assumed bounds $\tau_{\{v,w\}}$ on actual delays in our algorithm to ensure that communications between v and w started at a time $t - \tau_{\{v,w\}}$ induce communication updates at time t. Our algorithms thus behave exactly as if individual communication delays coincide with these upper bounds $\tau_{\{v,w\}}$, which allows us to analyze algorithms with constant, albeit heterogeneous delays.

In contrast an analysis in discrete time would use a global iteration counter, and discrete-time delays would not be constant,making the analysis either much more involved or unable to capture the asynchronous speedup described above.

Proof. Theorem 4.1 is obtained by applying a general result on delayed coordinate descent in the continuized framework that we detail in Section 4.3. Specifically, we consider the function $g(\lambda) = \frac{1}{2}\|A\lambda\|^2$ for $\lambda \in \mathbb{R}^{\mathcal{E}\times d}$ and $A \in \mathbb{R}^{\mathcal{V}\times\mathcal{E}}$ as defined in Section 4.2.1. As in Section 4.2.4, there exists $\lambda \in \mathbb{R}^{\mathcal{E}\times d}$ such that $x_0 - \bar{x} = A\lambda$. Let $(\lambda_t)_{t\geq 0}$ be defined with $\lambda_0 = \lambda$, and the delayed coordinate gradient steps at the clock tickings of the *P.p.p.*'s:

$$\lambda_{T_k} \leftarrow \lambda_{T_k-} - \frac{K_{\{v_k,w_k\}}}{p_{\{v_k,w_k\}}}\nabla_{\{v_k,w_k\}}g(\lambda_{T_k-\tau_{\{v_k,w_k\}}})\,.$$

For all $t \geq 0$, we then have $x_t = \bar{x} + A\lambda_t$, where we recall that the process (x_t) follows the delayed randomized gossip updates (4.10) of Algorithm 4.1. Then, for all $t \geq 0$, we have $g(\lambda_t) = \frac{1}{2}\|A\lambda_t\|^2 = \frac{1}{2}\|x_t - \bar{x}\|^2$.

The result of Theorem 4.1 follows from a control of $\mathbb{E}\left[g(\lambda_t)\right]$ that is a direct consequence of Theorem 4.2 in next section with the specific choices $m = |\mathcal{E}|$ and coordinate blocks corresponding to edges. The assumptions of Theorem 4.2 are verified with $L_{\{v,w\}} = 2\mu^2_{\{v,w\}}$, $M_{\{v,w\},\{v',w'\}} = \sqrt{L_{\{v,w\}}L_{\{v',w'\}}}$, and strong convexity parameter $\lambda_2(\Delta_G(\nu_{\{v,w\}} = \mu^2_{\{v,w\}}))$ for the specific choice $\mu^2_{\{v,w\}} = K_{\{v,w\}}$, as is shown in Lemmas 4.A.1, 4.A.2, 4.A.3 in the Appendix, giving us exactly Theorem 4.1. □

4.3. Delayed coordinate gradient descent in the continuized framework

Let J be a σ-strongly convex function on $\mathbb{R}^D$. For $k = 1, ..., m$, let E_k be a subspace of $\mathbb{R}^D$, and assume that:

$$\mathbb{R}^d = \bigoplus_{k=1}^{m} E_k\,, \tag{4.16}$$

where $\oplus$ denotes a direct sum of linear spaces. For $x \in \mathbb{R}^D$, let x_k denote its orthogonal projection on E_k and let $\nabla_k J := (\nabla J)_k$, and assume that the subspaces $E_1, ..., E_m$ are orthogonal. For $k, \ell \in [m]$, we say that k and ℓ are **adjacent** and we write $k \sim \ell$ if and only if $\nabla_k\nabla_\ell J = \nabla^2_{kl}J$ is not identically constant equal to 0. This induces a symmetric graph structure on the coordinates $k \in [m]$. In the context of gossip network averaging, $m = |\mathcal{E}|$ and each subspace E_k corresponds to an edge $e_k = \{v_k, w_k\}$ of the graph; in that context, we have $k \sim \ell$ if and only if edges e_k and e_ℓ share a node. In the network averaging problem previously described, the function J used is $g(\lambda) = \frac{1}{2}\|A\lambda\|$ for $\lambda \in \mathbb{R}^{\mathcal{E}\times d}$ the edge variables. Subspaces are $E_{\{v,w\}}$ of dimension d for $\{v, w\} \in \mathcal{E}$ (and $m = |E|$) corresponding to variables of λ associated to edge $\{v, w\}$.

4.3.1. Algorithm and assumptions

Continuized delayed coordinate gradient descent algorithm. For $k \in [m]$, let $\mathcal{P}_k$ be a *P.p.p.* of intensity p_k denoting the times at which an update can be performed on subspace E_k. For $t \in \mathcal{P}_k$ let $\varepsilon_k(t) \in \{0,1\}$ be the indicator of whether the update is performed or not. Let also η_k be some positive step size for $k \in [m]$. Consider then the following continuous-time process $X(t)$, where $X_k(t)$ (the projection of $X(t)$ on E_k) evolves according to:

$$\mathrm{d}X_k(t) = -\varepsilon_k(t)\eta_k \nabla_k J((X(t-\tau_k))\mathcal{P}_k(\mathrm{d}t)\,, \tag{4.17}$$

where $\mathcal{P}_k(\mathrm{d}t)$ corresponds to a Dirac at the points of the *P.p.p.* $\mathcal{P}_k$. In words, $(X(t))_{t\geq 0}$ is a jump process that takes coordinate gradient descent steps along subspaces $(E_k)_{k\in[m]}$ at the times of independent Poisson point processes $(\mathcal{P}_k)_{k\in[m]}$. We introduced variables $(\varepsilon_k(t))_{k\in[m],t\in\mathcal{P}_k}$ with values in $\{0,1\}$ to represent capacity constraints: $\varepsilon_k(t) = 0$ if the update at time $t \in \mathcal{P}_k$ cannot be performed due to some constraint saturation; these variables $\varepsilon_k(t)$ will be essential in our treatment of communication and computation capacity constraints in Section 4.5.

Regularity assumptions. J is σ-strongly convex, and L_k-smooth on E_k for $k \in [m]$. Furthermore, there exist non-negative real numbers $M_{k,\ell}$ and $M_{\ell,k}$ for $k \sim \ell$ such that for all $k = 1, ..., m$ and $x, y \in \mathbb{R}^D$, we have:

$$\|\nabla_k J(x) - \nabla_k J(y)\| \leq \sum_{\ell\sim k} M_{k,\ell}\|x_\ell - y_\ell\|\,. \tag{4.18}$$

When J is L_k smooth on E_k as we assume, the above condition is verified by the choice $M_{i,j} = L_w$, $i \sim j$. If $\nabla_k J$ is M_k-Lipschitz, Condition (4.18) is verified by the choice $M_{k,\ell} = M_k$. Assumption (4.18) however allows for more freedom, and is particularly well suited for our analysis. In particular for decentralized optimization, it will be convenient to take $M_{k\ell} = \sqrt{L_k L_\ell}$.

Assumptions on variables $\varepsilon_k(t)$, $t \in \mathcal{P}_k$. For $t \in \mathcal{P}_k$, random variable $\varepsilon_k(t)$ is $\sigma\big(\mathcal{P}_\ell \cap [t - \tau_k, t), \ell \in [m]\big)$-measurable, and there exists a constant $\varepsilon_k > 0$ such that:

$$\mathbb{E}\left[\varepsilon_k(t)\right] \geq \varepsilon_k\,,$$

Furthermore, we assume that $\varepsilon_k(t)$ is negatively correlated with each quantity $N_\ell(t-\tau_k, t) = |\mathcal{P}_\ell \cap [t - \tau_k, t]|$, *i.e.* that for all $k, \ell \in [m]$,

$$\mathbb{E}\left[\varepsilon_k(t)N_\ell(t-\tau_k,t)\right] \leq \mathbb{E}\left[\varepsilon_k(t)\right]\mathbb{E}\left[N_\ell(t-\tau_k,t)\right]. \tag{4.19}$$

In our subsequent treatment of communication and capacity contraints, we shall see that the above assumptions are verified for $\epsilon_k(t)$ the indicator that t is a point a *truncated P.p.p.* $\tilde{\mathcal{P}}_k$ defined as follows:

Definition 4.3.1 (Truncated P.p.p.). *Let $(\mathcal{P}_k)_{1\leq k\leq m}$ be* P.p.p. *of respective intensities $(p_k)_{1\leq k\leq m}$, $(\tau_k)_{1\leq k\leq m}$ non-negative delays. Let N_k be the Poisson point measures associated to $\mathcal{P}_k$, $k \in [m]$. For $(\mathcal{C}_r)_{1\leq r\leq M}$ subsets of $[m]$, we define the truncated Poisson point measures $(\tilde{N})_{1\leq k\leq m}$ of intensities $(p_k)_{1\leq k\leq m}$ and parameters $(\tau_k)_k, (q_{k,r})_{k\in[m],r\in[M]}$ as:*

$$\mathrm{d}\tilde{N}_k(t) = 1_{\left\{\bigcap_{1\leq k\leq M}\left\{\sum_{\ell\in\mathcal{C}_r} N_\ell([t-\tau_k,t))\leq q_{k,r}\right\}\right\}}\mathrm{d}N_k(t)\,, \tag{4.20}$$

and we let $\tilde{\mathcal{P}}_k$ be the point process associated to this point measure.

4.3.2. Convergence guarantees and analysis

The main result of this Section is the following

Theorem 4.2 (**Delayed Coordinate Gradient Descent**). *Under the stated assumptions on regularity of J and on variables $\epsilon_k(t)$, assume further that the step sizes η_k are given by $\eta_k = \frac{K_k}{p_k L_k}$ where for all $k \in [m]$,*

$$K_k \leq \frac{p_k}{1 + \sum_{\ell \sim k} p_\ell \left(\frac{\tau_k M_{k,\ell} + e\tau_\ell M_{\ell,k}}{\sqrt{L_k L_\ell}} \right)}, \tag{4.21}$$

and let $\gamma \in \mathbb{R}_+$ be such that:

$$\gamma < \min \left(\sigma \min_k \frac{\varepsilon_k K_k}{L_k}, \frac{1}{\tau_{\max}} \right), \tag{4.22}$$

where $\tau_{\max} := \max_{k \in [m]} \tau_k$. Then for any $T > 0$ the solution $X(t)$ to Equation (4.17) verifies

$$\frac{\int_0^T e^{\gamma t} \mathbb{E}\left[J(X(t)) - J(x^\star)\right] \mathrm{d}t}{\int_0^T e^{\gamma t} \big(J(X(0)) - J(x^\star)\big) \mathrm{d}t} \leq e^{-\frac{\gamma T}{2}} \frac{1 + \frac{\tau_{\max}}{T}}{1 - \gamma \tau_{\max}}. \tag{4.23}$$

Proof. We proceed in three steps. The first step consists in upper bounding, for $t \geq 0$, the quantity $\frac{\mathrm{d}\mathbb{E}[J(X(t))]}{\mathrm{d}t}$. We then introduce in Step 2 a Lyapunov function inspired by the Lyapunov-Krasovskii functional [GL09]), and by using the result proved in the first step, we show that it verifies a delayed ordinary differential inequality. The last step then consists in deriving the desired result from this delayed differential inequality.

Step 1 To bound $\frac{\mathrm{d}\mathbb{E}[J(X(t))]}{\mathrm{d}t}$, we study infinitesimal increments between t and $t + \mathrm{d}t$ for $\mathrm{d}t \to 0$. This approach is justified by results on *stochastic ordinary differential equation with Poisson jumps*, see [Dav84]. For $t \geq 0$, let $\mathcal{F}_t$ be the filtration induced by $\mathcal{P}_k \cap [0, t)$, $k \in [m]$ *i.e.*, the filtration up to time t. By convention, for non-positive t, we write $X(t) = X(0)$. The following inequalities are written up to $o(\mathrm{d}t)$ terms, that we omit to lighten notations. Finally, we write

$$g_{k,t} = \nabla_k J(X(t)), \quad k \in [m], t \geq 0.$$

We have, using local smoothness properties of J and the fact that for a *P.p.p.* $\mathcal{P}$ of intensity p, $\mathbb{P}(\mathcal{P} \cap [t, t + \mathrm{d}t] = \emptyset) = (1 - p)\mathrm{d}t + o(\mathrm{d}t)$ and $\mathbb{P}(\#\mathcal{P} \cap [t, t + \mathrm{d}t] = 1) = p\mathrm{d}t + o(\mathrm{d}t)$:

$$\begin{aligned}
&\frac{\mathbb{E}\left[J(X(t + \mathrm{d}t)) - J(X(t)) | \mathcal{F}_t\right]}{\mathrm{d}t} \\
&= \sum_{k=1}^m p_k \left(J\left(X(t) - \frac{\varepsilon_k(t) K_k}{p_k L_k} g_{k,t-\tau_k} \right) - J(X(t)) \right) \\
&\leq \sum_{k=1}^m p_k \left(-\frac{K_k}{p_k L_k} \langle \varepsilon_k(t) g_{k,t-\tau_k}, g_{k,t} \rangle \right. \\
&\left. + \frac{L_k}{2} \left\| \varepsilon_k(t) \frac{K_k}{p_k L_k} \nabla_k g_{k,t-\tau_k} \right\|^2 \right).
\end{aligned}$$

First, we rewrite $-\frac{\varepsilon_k(t) K_k}{p_k L_k} \langle g_{k,t-\tau_k}, g_{k,t} \rangle$ as

$$-\frac{\varepsilon_k(t) K_k}{p_k L_k} \|g_{k,t-\tau_k}\|^2 - \frac{\varepsilon_k(t) K_k}{p_k L_k} \langle g_{k,t-\tau_k}, g_{k,t} - g_{k,t-\tau_k} \rangle,$$

and bound the second term there by

$$\begin{aligned}
&-\frac{\varepsilon_k(t)K_k}{p_kL_k}\langle g_{k,t-\tau_k}, g_{k,t}-g_{k,t-\tau_k}\rangle\\
&\leq \frac{\varepsilon_k(t)K_k}{p_kL_k}\|g_{k,t-\tau_k}\|\,\|g_{k,t}-g_{k,t-\tau_k}\|\\
&\leq \frac{K_k}{p_kL_k}\|\varepsilon_k(t)g_{k,t-\tau_k}\|\sum_{\ell\sim k}M_{k,\ell}\|X_\ell(t)-X_\ell(t-\tau_k)\|\,,
\end{aligned}$$

where we used the Cauchy-Schwarz inequality and then local Lipschitz property (4.18) of $\nabla_k G$. Writing

$$\begin{aligned}
&\|X_\ell(t)-X_\ell(t-\tau_k)\|\\
&=\left\|\int_{(t-\tau_k)^+}^{t}\frac{\varepsilon_\ell(s)K_\ell}{p_\ell L_\ell}g_{\ell,s-\tau_\ell}N_\ell(\mathrm{d}s)\right\|,
\end{aligned}$$

where N_ℓ is the Poisson point measure associated to $\mathcal{P}_\ell$, we have (where we use a triangle inequality for integrals):

$$\begin{aligned}
&\frac{K_kM_{k,\ell}}{p_kL_k}\mathbb{E}\left[\|\varepsilon_k(t)g_{k,t-\tau_k}\|\|X_\ell(t)-X_\ell(t-\tau_k)\|\right]\\
&\leq\mathbb{E}\left[\int_{(t-\tau_k)^+}^{t}M_{k,\ell}\frac{\varepsilon_k(t)K_k\varepsilon_\ell(s)K_\ell}{L_kp_kp_\ell L_\ell}\|g_{k,t-\tau_k}\|\|g_{\ell,s-\tau_\ell}\|N_\ell(\mathrm{d}s)\right]\\
&\leq\mathbb{E}\left[\int_{(t-\tau_k)^+}^{t}\frac{1}{2}\left(\frac{K_k^2M_{k,\ell}}{p_k^2L_k\sqrt{L_kL_\ell}}\|\varepsilon_k(t)g_{k,t-\tau_k}\|^2\right.\right.\\
&\qquad\left.\left.+\frac{K_\ell^2M_{k,\ell}}{p_\ell^2L_\ell\sqrt{L_kL_\ell}}\|\varepsilon_\ell(s)g_{\ell,s-\tau_\ell}\|^2\right)N_\ell(\mathrm{d}s)\right].
\end{aligned}$$

For the first term, since both $\varepsilon_k(t)$ and $N_\ell(\mathrm{d}s)$ for s in the integral are independent from $X(t-\tau_k)$ (and thus from $g_{k,t-\tau_k}$), and where we write $N_\ell(u,v)$ the number of clock tickings of $\mathcal{P}_\ell$ in the interval $[u,v)$, we obtain:

$$\begin{aligned}
&\mathbb{E}\left[\int_{(t-\tau_k)^+}^{t}\frac{1}{2}\frac{K_k^2M_{k,\ell}}{p_k^2L_k\sqrt{L_kL_\ell}}\|\varepsilon_k(t)g_{k,t-\tau_k}\|^2\right]\\
&=\frac{\mathbb{E}\left[N_\ell(t-\tau_k,t)\varepsilon_k(t)\right]}{2}\frac{K_k^2M_{k,\ell}}{p_k^2L_k\sqrt{L_kL_\ell}}\mathbb{E}\left[\|g_{k,t-\tau_k}\|^2\right].
\end{aligned}$$

Furthermore, using our negative correlation assumption, $\mathbb{E}\left[N_\ell(t-\tau_k,t)\varepsilon_k(t)\right]\leq\mathbb{E}\left[N_\ell(t-\tau_k,t)\right]\mathbb{E}\left[\varepsilon_k(t)\right]=p_\ell\tau_k\mathbb{E}\left[\varepsilon_k(t)\right]$, and since $\varepsilon_k(t)$ and $g_{k,t-\tau_k}$ are independent, $\mathbb{E}\left[\varepsilon_k(t)\right]\mathbb{E}\left[\|g_{k,t-\tau_k}\|^2\right]=\mathbb{E}\left[\varepsilon_k(t)\|g_{k,t-\tau_k}\|^2\right]$.

For the second term, since the process $(\varepsilon_\ell(s)g_{\ell,s-\tau_\ell})_s$ is predictable (in the sense that it is independent from $N_u(\mathrm{d}s)$ for all u), we have

$$\begin{aligned}
&\mathbb{E}\left[\int_{(t-\tau_k)^+}^{t}\frac{K_\ell^2M_{k,\ell}}{2p_\ell^2L_\ell\sqrt{L_kL_\ell}}\|\varepsilon_\ell(s)g_{\ell,s-\tau_\ell}\|^2N_\ell(\mathrm{d}s)\right]\\
&=\int_{(t-\tau_k)^+}^{t}\frac{K_\ell^2M_{k,\ell}}{2p_\ell^2L_\ell\sqrt{L_kL_\ell}}\mathbb{E}\left[\|\varepsilon_\ell(s)g_{\ell,s-\tau_\ell}\|^2\right]\mathbb{E}\left[N_\ell(\mathrm{d}s)\right]\\
&=\int_{(t-\tau_k)^+}^{t}\frac{K_\ell^2M_{k,\ell}}{2p_\ell^2L_\ell\sqrt{L_kL_\ell}}\mathbb{E}\left[\|\varepsilon_\ell(s)g_{\ell,s-\tau_\ell}\|^2\right]p_\ell\mathrm{d}s\,.
\end{aligned}$$

Hence,

$$\begin{aligned}
&\frac{K_k M_{k,\ell}}{p_k L_k}\mathbb{E}\left[\|\varepsilon_k(t) g_{k,t-\tau_k}\| \|X_\ell(t) - X_\ell(t-\tau_k)\|\right] \\
&\leq \frac{p_\ell \tau_k K_k^2 M_{k,\ell}}{2p_k^2 L_k \sqrt{L_k L_\ell}}\mathbb{E}\left[\|\varepsilon_k(t) g_{k,t-\tau_k}\|^2\right] \\
&\quad + \int_{(t-\tau_k)^+}^{t} \frac{K_\ell^2 M_{k,\ell}}{2p_\ell^2 L_\ell \sqrt{L_k L_\ell}}\mathbb{E}\left[\|\varepsilon_\ell(s) g_{\ell,s-\tau_\ell}\|^2\right] p_\ell \mathrm{d}s\,.
\end{aligned}$$

Combining all our elements and taking $\mathrm{d}t \to 0$, we hence have:

$$\begin{aligned}
\frac{\mathrm{d}\mathbb{E}\left[J(X(t))\right]}{\mathrm{d}t} &\leq -\sum_{k=1}^{m} \frac{K_k}{L_k}\left(1 - \frac{K_k}{2p_k}\right)\mathbb{E}\left[\|\varepsilon_k(t) g_{k,t-\tau_k}\|^2\right] \\
&+ \sum_{k=1}^{m}\sum_{\ell\sim k} \frac{p_\ell \tau_k K_k^2 M_{k,\ell}}{2p_k L_k \sqrt{L_k L_\ell}}\mathbb{E}\left[\|\varepsilon_k(t) g_{k,t-\tau_k}\|^2\right] \\
&+ \sum_{k=1}^{m}\sum_{\ell\sim k} \int_{(t-\tau_k)^+}^{t} \frac{p_k K_\ell^2 M_{k,\ell}}{2p_\ell L_\ell \sqrt{L_k L_\ell}}\mathbb{E}\left[\|\varepsilon_\ell(s) g_{\ell,s-\tau_\ell}\|^2\right] \mathrm{d}s\,.
\end{aligned} \tag{4.24}$$

Step 2 Now, introduce the following Lyapunov function:

$$\mathcal{L}_T^\gamma = \int_0^T e^{\gamma t}\mathbb{E}\left[J(X(t)) - J(x^\star)\right]\mathrm{d}t\,,$$

that we wish to upper-bound by some constant, where γ is as in (4.22). We have:

$$\frac{\mathrm{d}\mathcal{L}_T^\gamma}{\mathrm{d}T} = J(X(0)) - J(x^\star) + \gamma\mathcal{L}_T^\gamma + \int_0^T e^{\gamma t}\frac{\mathrm{d}\mathbb{E}\left[J(X(t))\right]}{\mathrm{d}t}\mathrm{d}t\,.$$

Integrating the bound (4.24) on $\frac{\mathrm{d}\mathbb{E}[J(X(t))]}{\mathrm{d}t}$, we obtain, using $\int_0^T\int_{(t-\tau)^+}^t h(u)\mathrm{d}u\mathrm{d}t \leq \tau\int_0^T h(t)\mathrm{d}t$ for non-negative h:

$$\begin{aligned}
\frac{\mathrm{d}\mathcal{L}_T^\gamma}{\mathrm{d}T} &\leq J(X(0)) - J(x^\star) + \gamma\mathcal{L}_T^\gamma \\
&\quad - \sum_{k=1}^{m}\frac{K_k}{L_k}\left(1 - \frac{K_k}{2p_k}\right)\int_0^T e^{\gamma t}\mathbb{E}\left[\|\varepsilon_k(t) g_{k,t-\tau_k}\|^2\right]\mathrm{d}t \\
&\quad + \sum_{k=1}^{m} A_k \int_0^T e^{\gamma t}\mathbb{E}\left[\|\varepsilon_k(t) g_{k,t-\tau_k}\|^2\right]\mathrm{d}t\,,
\end{aligned}$$

where

$$A_k = \frac{K_k^2}{2p_k L_k}\sum_{\ell\sim k}\frac{p_\ell\tau_k M_{k,\ell}}{\sqrt{L_k L_\ell}} + e^{\gamma\tau_\ell}\frac{p_\ell\tau_\ell M_{\ell,k}}{\sqrt{L_k L_\ell}}\,.$$

Remark now that we have

$$\frac{K_k^2}{2p_k L_k} + A_k \leq \frac{K_k}{2L_k}, \quad k \in [m]. \tag{4.25}$$

Indeed, (4.25) is equivalent to

$$K_k \leq \frac{p_k}{1 + \sum_{\ell\sim k}\left(\frac{p_\ell\tau_k M_{k,\ell}}{\sqrt{L_k L_\ell}} + e^{\gamma\tau_\ell}\frac{p_\ell\tau_\ell M_{\ell,k}}{\sqrt{L_k L_\ell}}\right)}\,,$$

which follows from the assumed bounds (4.21) on K_k and the fact that $\gamma \leq 1/\tau_{\max}$, assumed in (4.22). we then have, using (4.25) and the fact that, by strong convexity, $J(X(t))-J(x^\star) \leq \frac{1}{2\sigma}\|\nabla J(X(t))\|^2 = \frac{1}{2\sigma}\sum_{k=1}^m \|\nabla g_{k,t}\|^2$:

$$\begin{aligned}
\frac{\mathrm{d}\mathcal{L}_T^\gamma}{\mathrm{d}T} &\leq J(X(0)) - J(x^\star) + \gamma\mathcal{L}_T^\gamma \\
&\quad - \sum_{k=1}^m \frac{K_k}{2L_k}\int_0^{T-\tau_k} e^{\gamma(t+\tau_k)}\mathbb{E}\left[\|\varepsilon_k(t+\tau_k)g_{k,t}\|^2\right]\mathrm{d}t \\
&\leq J(X(0)) - J(x^\star) + \gamma\mathcal{L}_T^\gamma \\
&\quad - \min_{k\in[m]}\left(\frac{K_k\varepsilon_k e^{\gamma\tau_k}}{2L_k}\right)\int_0^{T-\tau_{\max}} e^{\gamma t}\mathbb{E}\left[\sum_{k=1}^m \|g_{k,t}\|^2\right]\mathrm{d}t \\
&\leq J(X(0)) - J(x^\star) + \gamma\left(\mathcal{L}_T^\gamma - \mathcal{L}_{T-\tau_{\max}}^\gamma\right),
\end{aligned}$$

where we used the assumption (4.22) that $\gamma \leq \sigma \min_{k\in[m]}\left(\frac{K_k\varepsilon_k}{L_k}\right)$.

Step 3 The proof is then concluded by using the following lemma, to control solutions of this delayed ordinary differential inequality.

Lemma 4.3.1. *Let $h : \mathbb{R} \to \mathbb{R}^+$ a differentiable function such that:*

$$\begin{aligned}
&\forall t \leq 0\,,\ h(t) = 0\,,\\
&\forall t \geq 0\,,\ h'(t) \leq a + b(h(t) - h(t-\tau))\,,
\end{aligned}$$

for some positive constants a, b, τ verifying $\tau b < 1$. Then:

$$\forall t \in \mathbb{R}\,,\quad h(t) \leq \frac{a(t+\tau)}{1-\tau b}\,.$$

Proof. Let $\delta(t) = h(t) - h(t-\tau)$. For any $t \geq 0$, we have:

$$\begin{aligned}
\delta(t) &= \int_{t-\tau}^t h'(s)\mathrm{d}s \\
&\leq \int_{t-\tau}^t (a + b\delta(s))\mathrm{d}s\,.
\end{aligned}$$

Let $c = \frac{\tau a}{1-\tau b}$ (solution of $x = \tau(a+bx)$) and $t_0 = \inf\{t > 0 | \delta(t) \geq c\} \in \mathbb{R}\cup\{\infty\}$. Assume that t_0 is finite. Then, $\delta(t) < c$ for $t < t_0$ and by continuity $\delta(t_0) = c$, so that:

$$\begin{aligned}
c = \delta(t_0) &\leq \int_{t_0-\tau}^{t_0} (a + b\delta(s))\mathrm{d}s \\
&< \int_{t_0-\tau}^{t_0} (a + bc)\mathrm{d}s = \tau(a+bc) = c\,,
\end{aligned}$$

as for all $s < t_0$, $\delta(s) < c$. This is absurd, and thus t_0 is not finite: $\forall t > 0, \delta(t) < c$, giving us for all $t \geq 0$ $h'(t) \leq a + bc$ and thus $h(t) \leq c(t+\tau)/\tau$. □

To conclude the proof of Theorem 4.2, we apply Lemma 4.3.1 to $h(T) = \mathcal{L}_T^\gamma$ with $a = J(X(0)) - J(x^\star)$, $b = \gamma$ and $\tau = \tau_{\max}$ to obtain that for all $T > 0$,

$$\mathcal{L}_T^\gamma \leq \big(J(X(0)) - J(x^\star)\big)\frac{T + \tau_{\max}}{1 - \tau_{\max}\gamma}\,.$$

The result of Theorem 4.2 follows by dividing this inequality by $\int_0^T e^{\gamma t}\mathrm{d}t = \frac{e^{\gamma T}-1}{\gamma}$:

$$\begin{aligned}\frac{\int_0^T e^{\gamma t}\mathbb{E}\left[J(X(t)) - J(x^\star)\right]\mathrm{d}t}{\int_0^T e^{\gamma t}\big(J(X(0)) - J(x^\star)\big)\mathrm{d}t} &\leq \frac{\gamma}{e^{\gamma T}-1}\frac{T+\tau_{\max}}{1-\tau_{\max}\gamma}\\ &= \frac{\gamma T}{e^{\gamma T}-1}\frac{1+\tau_{\max}/T}{1-\tau_{\max}\gamma}\\ &\leq e^{-\gamma T/2}\frac{1+\tau_{\max}/T}{1-\tau_{\max}\gamma},\end{aligned}$$

where we used that for $x \geq 0$, $\frac{e^x-1}{x} \geq e^{x/2}$. □

4.4. Extension to decentralized optimization

Using Theorem 4.2, we are now armed to generalize the delayed randomized gossip algorithm and analysis to more general settings. In this section we extend our results to decentralized optimization, going beyond the quadratic objective functions considered network averaging.

4.4.1. Delayed Decentralized Optimization

Consider the decentralized optimization problem (4.1). We make the following assumptions on the individual objective functions f_v therein :

$$\textit{each } f_v,\ v \in \mathcal{V},\ \textit{is } \sigma\textit{-strongly convex and } L\textit{-smooth,} \tag{4.26}$$

see [Bub15] and Section 1.6 for definitions. Let $f(z) := \sum_{v\in[n]} f_v(z)$ for $z \in \mathbb{R}^d$ and $F(x) = \sum_{v\in[n]} f_v(x_v)$ for $x = (x_1, \cdots, x_n) \in \mathbb{R}^{n\times d}$ where $x_v \in \mathbb{R}^d$ corresponds to node $v \in [n]$.

Definition 4.4.1 (Fenchel Conjugate)**.** *For any function $g : \mathbb{R}^p \to \mathbb{R}$, its* Fenchel conjugate *is denoted by g^* and defined on $\mathbb{R}^p$ by $g^*(y) = \sup_{x\in\mathbb{R}^p}\langle x, y\rangle - g(x) \in \mathbb{R}\cup\{+\infty\}$.*

Our algorithm for delayed decentralized optimization is built on delayed randomized gossip for network averaging, augmented with local computations. Each node $v \in \mathcal{V}$ keeps two local variables: the communication variable $x_v(t)$, used to run delayed randomized gossip, and a computation variable $y_v(t)$, used to make local computation updates in the following way.

1. **Local computations.** Each node v generates a *Poisson point process*

$$\mathcal{P}_v^{\mathrm{comp}} = \{T_1^{\mathrm{comp}}(v) < T_2^{\mathrm{comp}}(v), \ldots\}$$

of intensity p_v^{comp}. At the clock tickings $T_k^{\mathrm{comp}}(v)$, a local computation update is made corresponding to a computation started at a time $T_k^{\mathrm{comp}}(v) - \tau_v^{\mathrm{comp}}$, where τ_v^{comp} is the upper bound on the time to perform an elementary computation at node v, introduced in Assumption 4.1.1. Thus by assumption the computation started at time $T_k^{\mathrm{comp}}(v) - \tau_v^{\mathrm{comp}}$ is completed by time $T_k^{\mathrm{comp}}(v)$ so that the update can be performed at that time. The precise form of this update is given by Equation (4.30).

2. **Communications.** In parallel of these local computations, a *Delayed Randomized Gossip* is run on the graph. Dedicated *P.p.p.* $(\mathcal{P}_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$ with respective intensities $(p_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$ are associated to communication updates of all network edges, and used to perform updates as prescribed by Equation (4.10) in *Delayed Randomized Gossip*.

The resulting Delayed Decentralized Optimization algorithm, or *DDO* for short, is described in Algorithm 4.3 from a local viewpoint and is a combination of Algorithm 4.1 for

communication updates along edges $\{v,w\} \in \mathcal{E}$ with Algorithm 4.2 for local computation updates at nodes $v \in \mathcal{V}$.

From a global viewpoint, the algorithm is generated as follows. Let $\{T_k\}_{k\geq 0}$ be a *P.p.p.* process of intensity $\sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}} + \sum_{v\in\mathcal{V}} p_v^{\text{comp}}$. For all $k \geq 0$, at time k a communication or a computation update is performed. With probability proportional to $p_{\{v,w\}}$ the communication update (4.10) is performed along edge $\{v,w\}$ ($T_k = T_\ell(\{v,w\})$ in that case), with probability proportional to p_v^{comp} the computation update (4.30) is performed at node v ($T_k = T_\ell^{\text{comp}}(v)$ in that case).

4.4.2. Convergence guarantees

The process $(x(t), y(t)) \in \mathbb{R}^{2n\times d}$ defined by algorithm *DDO*, Algorithm 4.3, satisfies the following convergence guarantees that generalize Theorem 4.1 to decentralized optimization beyond the case of quadratic functions.

Theorem 4.3 (**Delayed Decentralized Optimization**). *Under the regularity assumptions* (4.26), *assume further that for all $v \in \mathcal{V}$ and $\{v,w\} \in \mathcal{E}$, we have:*

$$\begin{aligned} K_{\{v,w\}} &\leq \frac{p_{\{v,w\}}}{1 + \sum_{\{v',w'\}\sim\{v,w\}} p_{\{v',w'\}}\left(\tau_{\{v,w\}} + e\tau_{\{v',w'\}}\right)} \\ K_v^{\text{comp}} &\leq \frac{p_v^{\text{comp}}}{1 + \sum_{w\sim v} p_{\{v,w\}}\left(\tau_v^{\text{comp}} + e\tau_{\{v,w\}}\right)} \,. \end{aligned} \tag{4.27}$$

Let $\tau_{\max} := \max\left(\max_{\{v,w\}\in\mathcal{E}} \tau_{\{v,w\}}, \max_{v\in\mathcal{V}} \tau_v^{\text{comp}}\right)$. Then for $\gamma > 0$ such that

$$\gamma \leq \min\left(\frac{\sigma}{4L}\lambda_2\big(\Delta_G(K)\big)\,,\ \frac{1}{\tau_{\max}}\right)\,, \tag{4.28}$$

the process $(x(t), y(t))$ generated by DDO *satisfy*

$$\frac{\int_0^T e^{\gamma t}\mathbb{E}\left[\left\|\frac{\sigma}{2}x(t) - \bar{x}^\star\right\|^2\right]\mathrm{d}t}{\int_0^T e^{\gamma t}\left\|\frac{\sigma}{2}x(0) - \bar{x}^\star\right\|^2\mathrm{d}t} \leq e^{-\frac{\gamma T}{2}}\frac{L}{\sigma}\frac{1+\frac{\tau_{\max}}{T}}{1-\gamma\tau_{\max}}\,, \tag{4.29}$$

where $\bar{x}^\star = (x^\star, \dots, x^\star)^\top \in \mathbb{R}^{n\times d}$ for $x^\star$ minimizer of $f = \sum_v f_v$.

DDO is based on a dual formulation and uses an augmented graph representation introduced in [HBM20] to decouple computations from communications, as detailed in the proof. The dual gradient computations in Algorithm 4.2 can be expensive in general; they could be avoided by using a primal-dual approach for the computation updates [KGGR21a].

The convergence guarantees we obtain resemble classical ones: Interpreting γ as the reciprocal of the time scale for convergence, we recognize in its upper bound (4.28) an "optimization factor" $\kappa_{\text{comp}}^{-1} := \sigma/L$, and a "communication factor" $\kappa_{\text{comm}}^{-1} = \lambda_2\big(\Delta_G(K)\big)$. Our method is non-accelerated, so the computation factor κ_{comp}, the condition number of the optimization problem, is expected. The communication factor captures the delay heterogeneity in the graph as in *Delayed Randomized Gossip*, leading to the *asynchronous speedup* discussed in Section 4.2 after Theorem 4.1.

Previous approaches have considered accelerating decentralized optimization by obtaining $\sqrt{\kappa_{\text{comp}}}$ instead of κ_{comp} and/or $\sqrt{\kappa'_{\text{comm}}}$ instead of κ'_{comm} for κ'_{comm} a communication factor in the rate of convergence [SBB+17, KSR20, HBM19]. Our result yields a speedup of a different nature: we obtain a communication factor κ_{comm} that can be arbitrarily larger than previously considered κ'_{comm} for networks with huge delay heterogeneity.

Algorithm 4.2: Local computations, node v

Step size $K_v^{\text{comp}} > 0$
Initialization $x_0(v) = y_0(v) = 0$
Initialization $T_1^{\text{comp}}(v) \sim \text{Exp}(p_v^{\text{comp}})$
for $\ell = 1, 2, \ldots$ **do**
  $T_{\ell+1}^{\text{comp}}(v) = T_\ell^{\text{comp}}(v) + \text{Exp}(p_v^{\text{comp}})$.
end for
for $\ell = 1, 2, \ldots$ **do**
  At time $T_\ell^{\text{comp}}(v) - \tau_v^{\text{comp}}$, node v computes $g_v = \nabla \phi_v^*(y_v(T_\ell^{\text{comp}}(v) - \tau_v^{\text{comp}}))$ (takes a time less than τ_v^{comp}) and keeps $\hat{x}_v = x_v(T_\ell^{\text{comp}}(v) - \tau_v^{\text{comp}})$ in memory, where $\phi_v = f_v - \frac{\sigma}{4}\|.\|^2$.
  At time $T_\ell^{\text{comp}}(v)$,

$$\begin{aligned} y_v(T_\ell^{\text{comp}}(v)) &\overset{t}{\leftarrow} y_v(T_\ell^{\text{comp}}(v)-) - \frac{\sigma K_v^{\text{comp}}}{p_v^{\text{comp}}}\left(g_v - \hat{x}_v\right), \\ x_v(T_\ell^{\text{comp}}(v)) &\overset{t}{\leftarrow} x_v(T_\ell^{\text{comp}}(v)-) - \frac{K_v^{\text{comp}}}{2p_v^{\text{comp}}}\left(\hat{x}_v - g_v\right). \end{aligned} \tag{4.30}$$

end for

Algorithm 4.3: *DDO*

Node initializations $x_0(v) = y_0(v) = 0$, $v \in \mathcal{V}$
for $v \in \mathcal{V}$ and $\{v, w\} \in \mathcal{E}$, asynchronously, in parallel **do**
  Communication updates along edge $\{v, w\}$ according to Algorithm 4.1
  Local computation updates at node v according to Algorithm 4.2
end for
Output: $\frac{\sigma}{2} x_v(t)$ at time t and node v.

4.4.3. Proof of Theorem 4.3

Proof. Following the augmented graph approach [HBM20], for each "physical" node $v \in \mathcal{V}$, we associate a "virtual" node v^{comp}, corresponding to the computational unit of node v. We then consider the augmented graph $G^+ = (\mathcal{V}^+, \mathcal{E}^+)$, where $\mathcal{V}^+ = \mathcal{V} \cup \mathcal{V}^{\text{comp}}$ (for $\mathcal{V}^{\text{comp}} = \{v^{\text{comp}}, v \in \mathcal{V}\}$) and $\mathcal{E}^+ = \mathcal{E} \cup \mathcal{E}^{\text{comp}}$ (for $\mathcal{E}^{\text{comp}} = \{(v^{\text{comp}}), v \in \mathcal{V}\}$).

For $v \in \mathcal{V}$, function f_v is then split (using σ-strong convexity) into a sum of two $\sigma/2$-strongly convex functions: $f_v = \phi_v + \phi_{v^{\text{comp}}}$ where $\phi_{v^{\text{comp}}}(x_v) = f_v(x_v) - \frac{\sigma}{4}\|x_v\|^2$ and $\phi_v(x_v) = \phi_{\text{comm}}(x_v) = \frac{\sigma}{4}\|x_v\|^2$.

The optimization objective (4.1)

$$\min_{x_1 = \ldots = x_n} \frac{1}{n} \sum_{i=1}^n f_v(x_v), \quad x = (x_1, \ldots, x_n) \in \mathbb{R}^{\mathcal{V} \times d}$$

can then be rewritten as

$$\min_{x \in \mathbb{R}^{V^+}} \left\{ F(x) = \sum_{v \in \mathcal{V}} \phi_v(x_v) + \sum_{v^{\text{comp}} \in \mathcal{V}^{\text{comp}}} \phi_{v^{\text{comp}}}(x_{v^{\text{comp}}}) \right\},$$

under the constraint $x_v = x_w$ for $\{v, w\} \in \mathcal{E}^+$. This constraint can then be rewritten as

$A^\top x = 0$ for $A \in \mathbb{R}^{E^+ \times V^+}$ such that for all $\{v,w\} \in \mathcal{E}^+$, $Ae_{\{v,w\}} = \mu_{\{v,w\}}(e_v - e_w)$, as was done for network averaging, considering the augmented graph instead of the original graph. Using Lagrangian duality, denoting $F_A^*(\lambda) := F^*(A\lambda)$ for $\lambda \in \mathbb{R}^{E^+ \times d}$ where F^* is the Fenchel conjugate of F, we have:

$$\min_{x \in \mathbb{R}^{V^+ \times d}, x_v = x_w, \{v,w\} \in \mathcal{E}^+} F(x) = \max_{\lambda \in \mathbb{R}^{E \times d}} -F_A^*(\lambda).$$

Thus $F_A^*(\lambda)$ is to be minimized over the dual variable $\lambda \in \mathbb{R}^{E^+ \times d}$. The rest of the proof is divided in two steps: in the first, we derive the updates of the *DDO* algorithm from coordinate gradient descent steps on dual variables, and in the second step we apply Theorem 4.2 to prove rates of convergence for these coordinate gradient descent steps on function F_A^*.

The partial derivative of F_A^* with respect to coordinate $\{v,w\} \in \mathcal{E}^+$ of $\lambda \in \mathbb{R}^{E^+ \times d}$ reads:

$$\nabla_{\{v,w\}} F_A^*(\lambda) = \mu_{\{v,w\}}(\nabla \phi_v^*((A\lambda)_v) - \nabla \phi_w^*((A\lambda)_w))\,.$$

Consider then the following step of coordinate gradient descent for F_A^* on coordinate $\{v_k, w_k\} \in \mathcal{E}^+$ of λ, performed when edge $\{v_k, w_k\}$ is activated at iteration k (corresponding to time t_k):

$$\lambda_{t_k} = \lambda_{t_k -} - \frac{1}{(2\sigma^{-1})\mu_{\{v_k,w_k\}}^2} \nabla_{\{v_k,w_k\}} F_A^*(\lambda_{t_k - \tau_{\{v_k,w_k\}}})\,, \tag{4.31}$$

corresponding to an instantiation of delayed coordinate gradient descent in the continuized framework, on function F_A^*, for *P.p.p.* of intensities $(p_{\{v,w\}})$ for $\{v,w\} \in \mathcal{E}$ and p_v^{comp} for $(iv^{\text{comp}}) \in \mathcal{E}^{\text{comp}}$. Denoting $v_t = A\lambda_t \in \mathbb{R}^{V^+ \times d}$ for $t \geq 0$, we obtain the following formula for updating coordinates v_k, w_k of v when $\{v_k, w_k\}$ activated, *irrespectively of the choice of* $\mu_{\{v,w\}}$ *in matrix* A:

$$\begin{aligned} v_{t_k, v_k} &= v_{t_k -, v_k} - \frac{\nabla \phi_{v_k}^*(v_{t_k - \tau_{\{v_k,w_k\}}, v_k}) - \nabla \phi_{w_k}^*(v_{t_k - \tau_{\{v_k,w_k\}}, w_k})}{2\sigma^{-1}}\,, \\ v_{t_k, w_k} &= v_{t_k -, w_k} + \frac{\nabla \phi_{v_k}^*(v_{t_k - \tau_{\{v_k,w_k\}}, v_k}) - \nabla \phi_{w_k}^*(v_{t_k - \tau_{\{v_k,w_k\}}, w_k})}{2\sigma^{-1}}\,. \end{aligned} \tag{4.32}$$

Such updates can be performed locally at nodes v and w after communication between the two nodes (if $\{v,w\}$ is a 'physical edge'), or locally (if $\{v,w\}$ is 'virtual edge'). We refer in the sequel to this scheme as the Coordinate Descent Method. While $\lambda \in \mathbb{R}^{\mathcal{E} \times d}$ is a dual variable defined on the edges, $v \in \mathbb{R}^{n \times d}$ is also a dual variable, but defined on the nodes. The *primal surrogate* of v is defined as $x = \nabla F^*(v)$ *i.e.* $x_v = \nabla f_v^*(v_v)$ at node v. It can hence be computed with local updates on v. The decentralized updates of Algorithm 4.3 (computational updates in Algorithm 4.2, communication updates in Algorithm 4.1) are then direct consequences of Equation (4.32).

The last step of the proof consists in applying Theorem 4.2 in order to obtain Theorem 4.3. The function F_A^* we introduced satisfies the assumptions of Theorem 4.2 with coordinate blocks corresponding to edges E^+: The regularity assumptions are satisfied with smoothness parameter $L_{\{v,w\}} = 8\mu_{\{v,w\}}^2 \sigma^{-1}$ and local Lipschitz coefficients $M_{\{v,w\},\{v',w'\}} = \sqrt{L_{\{v,w\}} L_{\{v',w'\}}}$ for any $\{v,w\}, \{v',w'\} \in \mathcal{E}^+$, as shown in Lemmas 4.A.1 and 4.A.2 in the Appendix. F_A^* is moreover σ-strongly convex[6] with σ derived using Lemmas 4.A.3 and 4.A.4, and the weights associated to matrix A are chosen so that $\mu_{\{v,w\}}^2 = \frac{\varepsilon_{\{v,w\}} K_{\{v,w\}} \sigma}{2\mu_{\{v,w\}}^2}$.

Finally, the output of the algorithm at node v is the primal surrogate of variable $x_v(t)$ (associated to ϕ_v), which is equal to $\nabla \phi_v(x_v(t)) = \frac{\sigma}{2} x_v(t)$.

[6] In fact, it is strongly convex on the orthogonal of $\text{Ker} A$, which suffices for us to conclude since the dynamics are restricted to this subspace.

□

4.5. Handling communication and computation capacity limits

4.5.1. Communication and computation capacity constraints

A given node or edge in the network may be able to handle only a limited number of communications or computations simultaneously. In Delayed Randomized Gossip and *DDO* algorithms, such constraints could be violated when some *P.p.p.* generates many points in a short interval. We extend our algorithms and resulting convergence guarantees to take into account these additional constraints.

In the continuized framework, this constraint can be enforced by truncating the *P.p.p.* that handles activations (Definition 4.3.1). We formalize communication and capacity constraints in Assumption 4.5.1, and show that asynchronous speedup is still achieved in this setting in Theorem 4.4.

In the previous sections, step size parameters $K_{\{v,w\}}, K_v^{\text{comp}}$ of the algorithms could be tuned to counterweight the effect of delays for arbitrary intensities $p_{\{v,w\}}$. With the introduction of capacity constraints we will see that the local optimizers at every node must also bound the intensities $p_{\{v,w\}}, p_v^{\text{comp}}$ based on local quantities. The resulting rate of convergence is the same as in Theorems 4.1 and 4.3, up to a constant factor of $1/2$.

We formalize communication and computation capacity constraints as follows.

Assumption 4.5.1 (Capacity constraints)**.** *For some $q_{\{v,w\}}$, q_v^{comm}, $q_v^{\text{comp}} \in \mathbb{N}^* \cup \{\infty\}$, $v \in \mathcal{V}$ and $\{v,w\} \in \mathcal{E}$,*

1. Computation Capacity: *Node v can compute only q_v^{comp} gradients in an interval of time of length τ_v^{comp};*
2. Communication Capacity, edge-wise limitations: *Only $q_{\{v,w\}}$ messages can be exchanged simultaneously between adjacent nodes $i \sim j$ in an interval of time of length $\tau_{\{v,w\}}$;*
3. Communication Capacity, node-wise limitations: *Node v can only send q_v^{comm} messages in any interval of time of length $\tau_v^{\text{comm}} = \max_{w\sim v} \tau_{\{v,w\}}$.*

Taking into account these constraints in the analysis boils down to replacing *P.p.p.* processes $(\mathcal{P}_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$, $(\mathcal{P}_v^{\text{comp}})_{v\in\mathcal{V}}$ of intensities $(p_{\{v,w\}})$, (p_v^{comp}) in the *DDO* algorithm, by *truncated Poisson point processes* $(\tilde{\mathcal{P}}_{\{v,w\}}, \tilde{\mathcal{P}}_v^{\text{comp}})$ (see Definition 4.3.1).

More precisely, for every edge $\{v,w\} \in \mathcal{E}$ (*resp.* node $v \in \mathcal{V}$), let $n_{\{v,w\}}(t)$ be the number of communications occurring along $\{v,w\}$ between times $t - \tau_{\{v,w\}}$ and t (*resp.* $n_{i,j}^{\text{comm}}$ the number of communications node v is involved in between times t and $t - \tau_{\{v,w\}}$, n_v^{comp} the number of computations node v is involved with between times t and $t - \tau_v^{\text{comp}}$). Without capacity constraints, these quantities are discrete Poisson random variables (of mean $p_{\{v,w\}}\tau_{\{v,w\}}$ for $n_{\{v,w\}}(t)$, *e.g.*).

4.5.2. Convergence guarantees

As in Section 4.4, we consider communication and computation update rules as in Algorithm 4.3 (*DDO* algorithm). In the presence of capacity constraints, a communication alongside edge $\{v,w\} \in \mathcal{E}$ at a clock ticking $t \in \mathcal{P}_{\{v,w\}}$ occurs and *does not break the communication capacity constraints* if and only if $n_{\{v,w\}}(t) < q_{\{v,w\}}$ (for edge-wise limitations), $n_{v,w}^{\text{comm}}(t) < q_v^{\text{comm}}$ and $n_{w,v}^{\text{comm}}(t) < q_w^{\text{comm}}$ (for node-wise limitations) are satisfied.

Under capacity constraints, we have the following guarantees for our algorithm, defined as in Algorithm 4.3 (Algorithm 4.1 for communications and Algorithm 4.2 for local computations), where communications and computations that violate the capacity constraints are dropped.

Theorem 4.4. *Assume for any $v \in \mathcal{V}$ and $\{v,w\} \in \mathcal{E}$:*

$$\begin{aligned} &cp_v^{\mathrm{comp}}\tau_v^{\mathrm{comp}} \leq q_v^{\mathrm{comp}}, \\ &cp_{\{v,w\}}\tau_{\{v,w\}} \leq q_{\{v,w\}}, \\ &c\sum_{w\sim v} p_{\{v,w\}}\tau_v^{\mathrm{comm}} \leq q_v^{\mathrm{comm}}, \end{aligned} \tag{4.33}$$

where $c = 1/(1-\sqrt{\ln(6)/2})$ is a numerical constant. Then, if the assumptions of Theorem 4.3 described in Equation 4.27 additionally hold, for γ verifying

$$\gamma \leq \min\left(\frac{\sigma}{8L}\lambda_2\big(\Delta_G(\nu_{\{v,w\}} = K_{\{v,w\}})\big)\,,\ \frac{1}{\tau_{\max}}\right),$$

we have:

$$\frac{\int_0^T e^{\gamma t}\mathbb{E}\left[\left\|\frac{\sigma}{2}x(t) - \bar{x}^\star\right\|^2\right]\mathrm{d}t}{\int_0^T e^{\gamma t}\left\|\frac{\sigma}{2}x(0) - \bar{x}^\star\right\|^2\mathrm{d}t} \leq e^{-\frac{\gamma T}{2}}\frac{L}{\sigma}\frac{1+\frac{\tau_{\max}}{T}}{1-\gamma\tau_{\max}}.$$

The same guarantees as without the capacity constraints thus hold, up to a constant factor $1/2$ in the rate of convergence. The conditions on the activation intensities (4.33) suggest that graph sparsity is beneficial: for q_v^{comm} small, $2\sum_{w\sim v} p_{\{v,w\}}\tau_v^{\mathrm{comm}} \leq q_v^{\mathrm{comm}}$ translates into $p_{\{v,w\}}$ scaling with the inverse of the edge-degree of $\{v,w\}$, so large degrees thus slow down the convergence. The new conditions (4.33) are easily enforced with the natural choice of intensities $p_{\{v,w\}}$ (*resp.* p_v^{comp}) of order $1/\tau_{\{v,w\}}$ (*resp.* τ_v^{comp}).

Taking $q_v^{\mathrm{comm}} = 1$, we recover the behavior of *loss networks* [Kel91a], where a node cannot concurrently communicate with different neighbors. Gossip on loss networks was previously studied in [EHM20], to obtain some form of asynchronous speedup. Comparatively, our present algorithms are structurally simpler and their analysis in the continuized framework yields faster convergence speeds.

4.5.3. Proof of Theorem 4.4

Proof. The algorithm under capacity constraints is obtained by applying coordinate gradient descent in the continuized framework to the same dual problem as in Section 4.4, but with random variables "$\varepsilon_k(t)$" that are not taken constant equal to 1. Here, for $\{v,w\} \in \mathcal{E}$ and $t \in \mathcal{P}_{\{v,w\}}$, we have

$$\varepsilon_{\{v,w\}}(t) = 1_{\left\{n_{\{v,w\}}(t)<q_{\{v,w\}}\,,\,n_v^{\mathrm{comm}}(t)<q_v^{\mathrm{comm}}\,,\,n_w^{\mathrm{comm}}(t)<q_w^{\mathrm{comm}}\right\}},$$

while for $v \in \mathcal{V}$ and $t \in \mathcal{P}_v^{\mathrm{comp}}$,

$$\varepsilon_{v\mathrm{comp}}(t) = 1_{\left\{n_v^{\mathrm{comp}}<q_v^{\mathrm{comp}}\right\}}.$$

We apply Theorem 4.2 as in the proof of Theorem 4.3, leading to the same stability conditions on the step sizes $K_{\{v,w\}}, K_v^{\mathrm{comp}}$, while the rate of convergence is multiplied by a lower bound ε on all $\mathbb{E}\left[\varepsilon_{\{v,w\}}(t)\right]$ and $\mathbb{E}\left[\varepsilon_{\{v,v^{\mathrm{comp}}\}}(t)\right]$. Let us finally compute such a lower bound ε.

For $\{v,w\} \in \mathcal{E}$, $n_{\{v,w\}}(t)$ is stochastically dominated by $Z_{\{v,w\}}$ a Poisson random variable of parameter $p_{\{v,w\}}\tau_{\{v,w\}}$, while $n_v^{\mathrm{comm}}(t)$ and $n_w^{\mathrm{comm}}(t)$ are respectively dominated by Z_v and Z_w, Poisson random variables of parameters $\tau_{\{v,w\}}\sum_{u\sim v} p_{\{u,v\}}$ and $\tau_{\{v,w\}}\sum_{u\sim w} p_{\{u,w\}}$, so that:

$$\begin{aligned} \mathbb{E}\left[\varepsilon_{\{v,w\}}(t)\right] &\geq \mathbb{P}\left(Z_{\{v,w\}} < q_{\{v,w\}}\,,\ Z_v < q_v^{\mathrm{comm}}\,,\ Z_w < q_w^{\mathrm{comm}}\right) \\ &\geq 1 - \mathbb{P}(Z_{\{v,w\}} \geq q_{\{v,w\}}) - \mathbb{P}(Z_v \geq q_v^{\mathrm{comm}}) - P(Z_w \geq q_w^{\mathrm{comm}}). \end{aligned}$$

We now prove that $\mathbb{P}(Z_{\{v,w\}} \geq q_{\{v,w\}}), \mathbb{P}(Z_v \geq q_v^{\text{comm}}), P(Z_w \geq q_w^{\text{comm}})$ are all inferior to $1/6$. For $\mathbb{P}(Z_{\{v,w\}} \geq q_{\{v,w\}})$, using that for Z a Poisson variable of parameter μ and $x \geq 0$,

$$\mathbb{P}(Z_\mu \geq \mu + x) \leq e^{\frac{-x^2}{\mu+x}},$$

we have $\mathbb{P}(Z_{\{v,w\}} \geq q_{\{v,w\}}) \leq e^{-\frac{(q_{\{v,w\}} - p_{\{v,w\}}\tau_{\{v,w\}})^2}{q_{\{v,w\}}}} \leq e^{-2(1-1/c)^2}$ if $q_{\{v,w\}} \geq 2$, and this quantity is equal to $1/6$, by definition of c. Then, if $q_{\{v,w\}} = 1$, using $\mathbb{P}(Z_\mu \geq 1) = 1 - e^{-\mu}$, we have that $\mathbb{P}(Z_{\{v,w\}} \geq q_{\{v,w\}}) \leq p_{\{v,w\}}\tau_{\{v,w\}} \leq 1/c \leq 1/6$. We proceed in the same way for $\mathbb{P}(Z_v \geq q_v^{\text{comm}}), P(Z_w \geq q_w^{\text{comm}})$. Hence, $\mathbb{E}\left[\varepsilon_{\{v,w\}}(t)\right] \geq 1/2$ under our assumptions on the Poisson intensities. Similarly, we prove that $\mathbb{E}\left[\varepsilon_{\{v,v^{\text{comp}}\}}(t)\right] \geq 1/2$, and this concludes the proof. □

4.6. Braess's Paradox and Experiments

In this section, we investigate how the local step sizes $K_{\{v,w\}}$ and Poisson intensities $p_{\{v,w\}}$ used in Theorems 4.1, 4.3 and 4.4 should be tuned for a fixed choice of communication delays. Consider the line graph with constant delays $\tau_{v,v+1} = \tau$. Add edge $(1, n)$ in order to close the line, with a delay $\tau_{1n} = \tau'$ with arbitrarily large τ'. If the added Poisson intensity p_{1n} satisfies $\tau_{1n}p_{1n} \to \infty$, then according to Theorem 4.1, we have $K_{12} \to 0$ and $K_{n-1,n} \to 0$. Consequently, since $\gamma = \mathcal{O}(\Delta_G(K))$, we have $\gamma \to 0$: the weighted graph becomes close to disconnected. By adding an edge to the graph, the convergence speed of delayed randomized gossip is degraded.

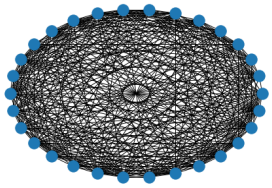
In order to alleviate the phenomenon, we would need to virtually delete the edge, by setting $p_{1n} = 0$. Figure 4.1 illustrates this phenomenon in the more general setting: one can sparsify the communication graph by solving a regularized optimization problem over the $p_{\{v,w\}}$ in order to maximize $\lambda_2(\Delta_G(K))$ (K being a function of p), leading to both faster consensus and smaller communication complexity (and thus lower energy footprint).

In road-traffic, removing one or more roads in a road network can speed up the overall traffic flow. This phenomenon, called *Braess's paradox* [EK10], also arises in loss networks [BKT97]. In our problem, this translates to removing an edge $\{v, w\}$ with a non-negligible Poisson intensity $p_{\{v,w\}}$. We take G_1 a dense Erdős-Rényi random graph (Figure 4.1a) of parameters $n = 30, p = 0.75$. Delays $\tau_{\{v,w\}}$ are taken equal to 0.01 with probability 0.9, and to 1 with probability 0.1. Initially, intensities are set as $p^{(1)}_{\{v,w\}} = 1/\tau_{\{v,w\}}$. Maximizing:

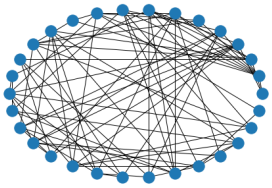
$$\lambda_2\left(\Delta_G\Big(\frac{p_{\{v,w\}}}{1 + \sum_{\{v',w'\}\sim\{v,w\}} p_{kl}(\tau_{\{v,w\}} + e\tau_{\{v',w'\}})}\Big)\right) - \omega \sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}}\tau_{\{v,w\}}$$

over $(p_{\{v,w\}})_{\{v,w\}}$, we obtain intensities $p^{(2)}$ and a graph G_2 (Figure 4.1b), sparser than G_1: we delete edges that have a null intensity (*i.e.* such that $p^{(2)}_{\{v,w\}} = 0$). We then run our delayed gossip algorithm for initialization x_0 a Dirac mass ($x_0(v) = I_{i=i_0}$), on G_1 (blue curves) and G_2, for the choice of $K_{\{v,w\}}$ as in Theorem 4.1. The green curve is the synchronous gossip algorithm [DKM+10] on G_1, to illustrate the asynchronous speedup, where each iteration takes a time $\tau_{\max} = 1$. In Figure 4.1d, the error to the consensus is measured as a function of the continuous time, while it is measured in terms of number of updates in Figure 4.1c and in terms of energy (defined as $\sum_{k:T_k<t} \tau_{\{v_k,w_k\}}$ at time t: the energy consumed by a communication is assumed to be proportional to the time the communication took) in Figure 4.1e.

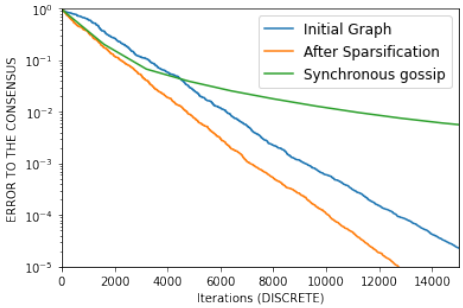
As expected, in terms of number of updates in the whole graph and energy spent, the sparser graph is more effective: slow and costly edges were deleted. Perhaps more surprising,



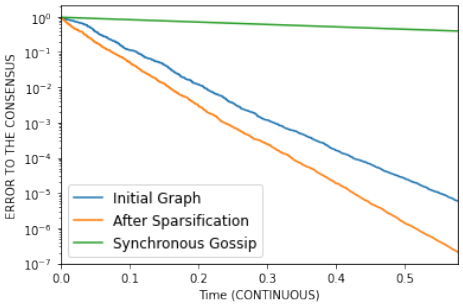
(a) – *Graph G_1 (E-R)*



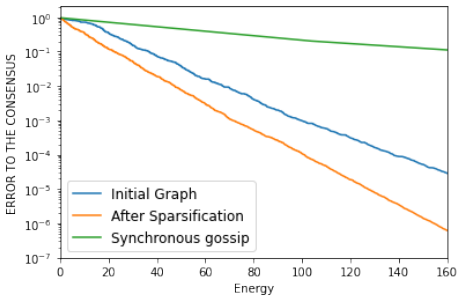
(b) – *Graph G_2 (after sparsification)*



(c) – *Continuous time*



(d) – *Number of updates in the whole graph*



(e) – *Energy spent*

Figure 4.1 – **Experiments and Braess's paradox.**

but supported by our theory (Theorem 4.3) and the resulting Braess's paradox, this also holds in Figure 4.1d: even though in the same amount of time, less updates are made in the sparser graph G_2 than in G_1, delayed randomized gossip is still faster on G_2 than G_1. Making less updates and deleting some communications make all other communications more efficient.

We believe that this phenomenon could be exploited for efficient design of large scale networks, beyond the maximization the spectral gap regardless of physical constraints as in [YYC+21] for instance.

Conclusion

We introduced in this chapter a novel analysis framework for the study of algorithms in the presence of delays, establishing that an *asynchronous speedup* can be achieved in decentralized optimization. Our results hold for explicit choices of algorithm parameters based on local network characteristics. They derive from the continuous-time analysis and assumptions handled in our continuized framework. The explicit conditions and convergence rates we obtain allow us to further discuss counter-intuitive effects akin to the Braess paradox, such as the possibility to speed up convergence by suppressing communication links. Although the algorithm requires dual updates, a fully primal algorithm could be obtained by using Bregman gradients [HBM20] or a primal-dual formulation [KSR20] or maybe more simply, by using the results from next chapter.

Appendix of Chapter 4

4.A. Additional technical lemmas

x

4.A.1. Regularity of F_A^*

For σ-strongly convex and L-smooth functions $f_1, \dots, f_n$ on $\mathbb{R}^d$ and for $A \in \mathbb{R}^{\mathcal{V}\times\mathcal{E}}$ such that $Ae_{\{v,w\}} = \mu_{\{v,w\}}(e_v - e_w)$ for $\{v,w\} \in \mathcal{E}$, define $F_A^* : \mathbb{R}^{\mathcal{E}\times d} \to \mathbb{R}$ as:

$$F_A^*(\lambda) = \frac{1}{n}\sum_{v\in\mathcal{V}} f_v((A\lambda_v)), \quad \lambda \in \mathbb{R}^{\mathcal{E}\times d}.$$

Lemma 4.A.1. *For any $\{v,w\} \in \mathcal{E}$, F_A^* is $L_{\{v,w\}} := 4\mu_{\{v,w\}}^2\sigma^{-1}$-smooth on $E_{\{v,w\}}$ the subspace of coordinates $\{v,w\} \in \mathcal{E}$.*

Proof. Let $h_{\{v,w\}} \in \mathbb{R}^d$ and $\lambda \in \mathbb{R}^{\mathcal{E}\times d}$. Using the σ^{-1}-smoothness of f_v^* and f_w^*:

$$\begin{aligned}
F_A^*(\lambda + e_{\{v,w\}}h_{\{v,w\}}^\top) - F_A^*(\lambda) &= f_v^*((A(\lambda + e_{\{v,w\}}h_{\{v,w\}})^\top)_v) - f_v^*((A\lambda)_v)\\
&+ f_w^*((A(\lambda + e_{\{v,w\}}h_{\{v,w\}}^\top))_w) - f_w^*((A\lambda)_w)\\
&\le \langle \nabla_{\{v,w\}}F_A^*(\lambda), e_{\{v,w\}}h_{\{v,w\}}^\top\rangle\\
&+ \frac{\sigma^{-1}}{2}\left\|(Ae_{\{v,w\}}h_{\{v,w\}}^\top)_v\right\|^2 + \frac{\sigma^{-1}}{2}\left\|(Ae_{\{v,w\}}h_{\{v,w\}}^\top)_w\right\|^2,
\end{aligned}$$

concluding the proof, as $\left\|(Ae_{\{v,w\}}h_{\{v,w\}}^\top)_v\right\|^2 = 2\mu_{\{v,w\}}^2\left\|h_{\{v,w\}}\right\|^2$. □

Lemma 4.A.2. *For any $\{v,w\} \in \mathcal{E}$, any $\lambda, \lambda' \in \mathbb{R}^{E^+\times d}$:*

$$\left\|\nabla_{\{v,w\}}F_A^*(\lambda) - \nabla_{\{v,w\}}F_A^*(\lambda')\right\| \le \sum_{\{v',w'\}\sim\{v,w\}} M_{\{v,w\},\{v',w'\}}\left\|\lambda_{kl} - \lambda'_{kl}\right\|, \tag{4.34}$$

where $M_{\{v,w\},\{v',w'\}} = \sqrt{L_{\{v,w\}}L_{\{v',w'\}}}$ and $L_{\{v,w\}} = 4\mu_{\{v,w\}}^2\sigma^{-1}, L_{\{v',w'\}} = 4\mu_{\{v',w'\}}^2\sigma^{-1}$.

Proof. Since $\nabla_{\{v,w\}}F_A^*(\lambda) = (Ae_{\{v,w\}})^\top((\nabla g_v^*((A\lambda)_v) - \nabla g_w^*((A\lambda)_w))$, we have:

$$\begin{aligned}
&\left\|\nabla_{\{v,w\}}F_A^*(\lambda) - \nabla_{\{v,w\}}F_A^*(\lambda')\right\|\\
&= \left\|(Ae_{\{v,w\}})^\top((\nabla f_v^*((A\lambda)_v) - \nabla f_v^*((A\lambda')_v)\right.\\
&\quad \left. - \nabla f_w^*((A\lambda)_w) + \nabla f_w^*((A\lambda')_w))\right\|\\
&\le \left\|Ae_{\{v,w\}}\right\|\left(\left\|\nabla f_v^*((A\lambda)_v) - \nabla f_v^*((A\lambda')_v)\right\|\right.\\
&\quad \left. + \left\|\nabla f_w^*((A\lambda)_w) - \nabla f_w^*((A\lambda')_w)\right\|\right)\\
&\le \sqrt{2}|\mu_{\{v,w\}}|\left(\sigma_v^{-1}\left\|(A\lambda)_v - (A\lambda')_v\right\| + \sigma_w^{-1}\left\|(A\lambda)_w - (A\lambda')_w\right\|\right)
\end{aligned}$$

$$
\begin{aligned}
&= \sqrt{2}|\mu_{\{v,w\}}| \left(\sigma_v^{-1} \left\| \sum_{u\sim v} \mu_{\{u,v\}} (\lambda_{\{u,v\}} - \lambda'_{\{u,v\}}) \right\| \right. \\
&\quad \left. + \sigma_w^{-1} \left\| \sum_{u\sim w} \mu_{\{u,w\}} (\lambda_{\{u,w\}} - \lambda'_{\{u,w\}}) \right\| \right) \\
&\leq \sqrt{2}|\mu_{\{v,w\}}| \left(\sigma_v^{-1} \sum_{u\sim v} |\mu_{\{u,v\}}| \left\| \lambda_{\{u,v\}} - \lambda'_{\{u,v\}} \right\|) \right. \\
&\quad \left. + \sigma_w^{-1} \sum_{u\sim w} |\mu_{\{u,w\}}| \left\| \lambda_{\{u,w\}} - \lambda'_{\{u,w\}} \right\|) \right)
\end{aligned}
$$

$$
\begin{aligned}
&\leq \sqrt{2}|\mu_{\{v,w\}}| \left(\sqrt{\sigma_v^{-1} + \sigma_w^{-1}} \sqrt{\sigma_v^{-1} + \sigma_k^{-1}} \sum_{u\sim v} |\mu_{\{u,v\}}| \left\| \lambda_{\{u,v\}} - \lambda'_{\{u,v\}} \right\|) \right. \\
&\quad \left. + \sqrt{\sigma_v^{-1} + \sigma_w^{-1}} \sqrt{\sigma_l^{-1} + \sigma_w^{-1}} \sum_{u\sim w} |\mu_{\{u,w\}}| \left\| \lambda_{\{u,w\}} - \lambda'_{\{u,w\}} \right\|) \right) \\
&\leq \sum_{\{v',w'\}\sim\{v,w\}} \sqrt{L_{\{v,w\}} L_{\{v',w'\}}} \left\| \lambda_{\{v',w'\}} - \lambda'_{\{v',w'\}} \right\|,
\end{aligned}
$$

where $L_{\{v,w\}}, L_{\{v',w'\}}$ as in Lemma 4.A.1. □

Lemma 4.A.3 (Strong convexity)**.** *The strong convexity parameter σ_A of F_A^* on the orthogonal of* $\ker(A)$ *is lower bounded by $L^{-1}\lambda_2(\Delta_G(\mu^2_{\{v,w\}}))$, where we recall that $\lambda_2(\Delta_G(\mu^2_{\{v,w\}}))$ is the graph Laplacian with weights $\mu^2_{\{v,w\}}$.*

Proof. Let $\lambda, \lambda' \in \mathbb{R}^{\mathcal{E}\times d}$. For $v \in \mathcal{V}$, by L^{-1}-strong convexity of f_v^*:

$$
\begin{aligned}
f_v^*((A\lambda)_v) - f_v^*((A\lambda')_v) \geq \langle \nabla f_v^*((A\lambda')_v), (A(\lambda - \lambda'))_v \rangle \\
+ \frac{1}{2L} \|(A(\lambda - \lambda'))_v\|^2.
\end{aligned}
$$

Summing over all $v \in \mathcal{V}$ and using $\nabla F_A^*(\lambda') = A^\top(\nabla_v f_v^*((A\lambda')_v))_{v\in\mathcal{V}}$ leads to:

$$
\begin{aligned}
F_A^*(\lambda) - F_A^*(\lambda') &\geq \langle \nabla F_A^*(\lambda'), \lambda - \lambda' \rangle + \frac{1}{2L} \|A(\lambda' - \lambda\|^2 \\
&\geq \langle \nabla F_A^*(\lambda'), \lambda - \lambda' \rangle + \frac{\lambda^+_{\min}(A^T A)}{4} \|\lambda - \lambda'\|^{*2}.
\end{aligned}
$$

where $\|.\|^*$ is the euclidean norm on the orthogonal of $Ker(A)$. Finally, notice that $AA^\top = \Delta_G(\mu^2_{\{v,w\}})$ and has same eigenvalues as $A^\top A$. □

4.A.2. The smallest positive eigenvalue of the augmented graph's weighted Laplacian matrix

Let $G = (V, E)$ be the "physical" graph, augmented as $G^+ = (V^+, E^+)$, where $V^+ = V \cup \{v^{\text{comp}}, v \in \mathcal{V}\}$ and $E^+ = E \cup \{(iv^{\text{comp}}), v \in \mathcal{V}\}$ as in Section 4.4.

Lemma 4.A.4. *For $\nu^+ = (\nu_{\{v,w\}})_{\{v,w\}\in\mathcal{E}^+}$ non negative weights, the smallest positive eigenvalue of the Laplacian of the augmented graph G^+ with weights ν^+ satisfies:*

$$
\lambda_2\big(\Delta_{G^+}(\nu^+)\big) \geq \frac{1}{4} \min\left(\lambda_2\big(\Delta_G(\nu)\big) , \min_{v\in\mathcal{V}} \nu_{iv^{\text{comp}}} \right),
$$

where $\lambda_2\big(\Delta_G(\nu)\big)$ is the smallest eigenvalue of the original graph, with weights $\nu = (\nu_{\{v,w\}})_{\{v,w\}\in\mathcal{E}}$.

Proof. Let $m = \min_{v\in\mathcal{V}} \nu_{iv^{\mathrm{comp}}}$ and $\lambda = \lambda_2\big(\Delta_G(\nu)\big)$. For any $X = (x, y) \in \mathbb{R}^{V^+}$, we have:

$$\begin{aligned} X^\top \Delta_{G^+}(\nu^+) X &= \sum_{\{v,w\}\in\mathcal{E}^+} \nu_{\{v,w\}}(X_v - X_w)^2 \\ &= x^\top \Delta_G(\nu) x + \sum_{v\in\mathcal{V}} \nu_{\{v,v^{\mathrm{comp}}\}}(x_v - y_v)^2 \\ &\geq \lambda \big\|x - \bar{X}\big\|^2 + m\|x - y\|^2 . \end{aligned}$$

Then, for $c > 0$ sufficiently small such that for any $z, z' \in \mathbb{R}$, $\lambda z^2 + m(z - z')^2 \geq cz^2 + cz'^2$, we have $X^\top \Delta_{G^+}(\nu^+) X \geq c\big\|X - \bar{X}\big\|^2$ and so $\lambda_2\big(\Delta_{G^+}(\nu^+)\big) \geq c$. Let us now compute such a value c, to conclude this proof.

For $z, z' \in \mathbb{R}$,

$$\begin{aligned} &\lambda z^2 + m(z - z')^2 - cz^2 + cz'^2 \\ &= (\lambda + m - c)z^2 + (m - c)z'^2 - 2mzz' \\ &= \left(\sqrt{\lambda + m - c}\, z - \frac{m}{\sqrt{\lambda + m - c}\, z'}\right) \\ &\quad + \left(m - c + \frac{m^2}{\lambda + m - c}\right) z' , \end{aligned}$$

and this quantity is non-negative as long as $c \leq \min(\lambda, m)/4$. □

Chapter 5

From Asynchronous SGD to Decentralized Asynchronous SGD

In previous chapters, we have studied asynchronous computations and communications, in the centralized setting (Chapter 3) and in the decentralized setting in the continuized framework (Chapter 4). However, this latter chapter suffers from practicability issues that we rightfully acknowledged: computations involve gradients of Fenchel conjugates. We attack the practicability issue in this chapter, by introducing algorithms that are as natural as possible, and analyze the corresponding sequence of iterates.

This chapter is inspired by [KLB+20] that provided a unified analysis of synchronous decentralized SGD algorithms, under generic communication schemes. We combine the ideas introduced by [KLB+20] with those of Chapter 3, in order to introduce a generic framework – that we call Asynchronous SGD on graphs – to study asynchronous and decentralized SGD algorithms. As opposed to the previous chapter, computations only involve local (delayed) SGD steps, while communications are first assumed to be non-delayed. However, the generality of our assumptions then allows us to handle communication delays through the introduction of loss network communication schemes, that prevent communication updates from overwriting over each other. The resulting algorithm enjoys an asynchronous speedup similar to that of Chapter 4, in the form of a graph eigengap with local weights only involving local delays.

Our general algorithmic framework covers asynchronous versions of many popular algorithms including SGD, Decentralized SGD, Local SGD, FedBuff [NMZ+22], thanks to its relaxed communication and computation assumptions. We provide rates of convergence under much milder assumptions than previous decentralized asynchronous works, while still recovering or even improving over the best know results for all the algorithms covered.

5.1. Introduction

We consider solving stochastic optimization problems that are distributed amongst n agents (indexed by a set $\mathcal{V}$) who can compute stochastic gradients in parallel. This includes classical federated setups, such as distributed and federated learning. Depending on the application, agents have access to either same shared data distribution or a different agent-specific distributions. In recent years, such stochastic optimization problems have continued to grow rapidly in size, both in terms of the dimension d of the optimization variable—i.e., the number of model parameters in machine learning—and in terms of the quantity of data—i.e., the number of data samples m being used over all agents. With d and m regularly reaching the hundreds or thousands of billions [Cc22, Tc23], it is increasingly necessary to use parallel optimization algorithms to handle the large scale.

With *communication cost* being one of the major bottlenecks of parallel optimization algorithms, there are several directions aimed to improve communication efficiency. Amongst the others (such as local update steps [Sti18, WPS+20a] and communication compression

[AGL+17, KSJ19]), **decentralization** and **asynchrony** are the two popular techniques for reducing the communication time. Decentralization [KLB+20, LZZ+17] eliminates the dependency on the central server—frequently a major bottleneck in distributed learning—while naturally amplifying privacy guarantees [CEBM22]. Asynchrony [RRWN11, Bau78, TBA86] shortens the time per computation rounds and allows more updates to be made during the same period of time. It aims to overcome several possible sources of delays: nodes may have *heterogeneous hardware* with different computational throughputs [KMA+19, HLA+21], *network latency* can slow the communication of gradients, and nodes may even just *drop out* [RGPP21]. Moreover, slower "*straggler*" compute nodes can arise in many natural parallel settings, including training ML models using multiple GPUs [CPM+16] or in the cloud; sensitivity to these stragglers poses a serious problem for synchronous algorithms, that depend on the slowest agent. In decentralized synchronous optimization where communication times between pairs of nodes may be heterogeneous, the algorithm can even be further slowed down by *straggling communication links*.

Combining both decentralization and asynchrony is a challenging problem, and it is only recently that this question has received attention [AR21, BRW+23, LHZQ20, LYW+22, EHM21a, ZY21, NSD+21]. These works are however restricted to a given communication protocol and static topologies [AR21, LHLL15, BRW+23, EHM21a, NSD+21], no communication delays [LHLL15, BRW+23, NSD+21], or their analyses rely on an upper-bound on the maximal computation delay [AR21, LHLL15, BRW+23, LHZQ20, LYW+22, NSD+21, ZY21, WLMJ23]. In this work we aim to circumvent these shortcomings. We study an asynchronous version of decentralized SGD in a unified framework that relaxes overly strong communication assumptions imposed by prior works. Our framework covers time-varying topologies, arbitrary computation orders and local update steps. We prove an improved rates of convergence under such a weaker communication assumptions, covering and improving asynchronous versions of many common distributed and federated algorithms.

5.1.1. Outline of this chapter

(i) We introduce **AGRAF SGD** (Asynchronous SGD on graphs), a unified formulation of an asynchronous version of the synchronous Decentralized SGD as formulated by [KLB+20]. One of the strengths of AGRAF SGD is that it formally takes the form of a simple sequence (Equation (5.3)), allowing for an effective theoretical analysis, while covering asynchronous versions of many distributed algorithms such as Asynchronous SGD, Decentralized SGD, FedAvg or FedBuff.

(ii) We analyze the AGRAF SGD sequence under various combinations of convexity, non-convexity, smoothness and Lispchitzness assumptions. We use a relaxed communication assumption that only imposes that the different topologies mix in a given window of time, while our computation assumption depends on whether the local functions are homogeneous or heterogeneous. In special cases, our rates recover best known rates of Minibatch SGD, Asynchronous SGD or Decentralized SGD, while for Asynchronous Decentralized SGD, our rates improve the previous works by up to factors of order n^2, under relaxed assumptions (as summarized in Table 5.1).

(iii) Finally, we show that AGRAF SGD allows to efficiently handle communication delays in decentralized optimization, by introducing *Decentralized SGD on Loss Networks*. We show that the assumptions required in our analysis are satisfied by this algorithm, giving explicit rates of convergence that depend on the underlying graph topology, pairwise communication delays, and each device computation time.

Related works specific to this chapter. Closely related to our analysis techniques, [MPP+17] proposed and utilized the analysis tool of **virtual iterates** for Asynchronous SGD under bounded delays, extended by [KSJ22, MBEW22] who proved that Asynchronous SGD performs well under arbitrary delays. We adapt this proof approach to decentralized optimiza-

tion in order to obtain some robustness towards large delays and introduce a different virtual sequence for the averaged model over all the nodes.

The closest asynchronous and decentralized works to ours [LHLL15, BRW+23] proposed asynchronous versions of decentralized SGD where at each iteration, **one** node v_k is sampled *independently from the past* (with fixed probabilities), and this node performs a local stochastic gradient step and an averaging operation with its neighbors. We extend their sequence and results to a more general (due to relaxed communication and computation assumptions) asynchronous version of decentralized SGD, that keeps the "unified" point of view of the work of [KLB+20].

[NBO23] considers the continuized framework of Chapter 2 for communication acceleration on time-varying topologies with local stochastic gradient steps: while their work is communication efficient, they do not consider computation or communication delays.

5.2. AGRAF Algorithmic Framework

In this section we present AGRAF SGD—our algorithmic framework for asynchronous decentralized SGD—and give examples of existing popular algorithms that it can cover.

5.2.1. Asychronous SGD on graphs

We consider a connected undirected graph $G = (\mathcal{V}, \mathcal{E})$[1] on a set of nodes $\mathcal{V} = \{1, \dots, n\}$. Let the function $f_v : \mathbb{R}^d \to \mathbb{R}$ of agent $v \in \mathcal{V}$ be defined as

$$f_v(x) := \mathbb{E}\left[F_v(x, \xi_v)\right] \quad \xi_v \sim \mathcal{D}_v\,, \quad x \in \mathbb{R}^d\,, \tag{5.1}$$

where $\mathcal{D}_v$ is some local distribution. Let the global objective function $f : \mathbb{R}^d \to \mathbb{R}$ be defined as follows, and consider the optimization problem

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) := \sum_{v \in \mathcal{V}} q_v f_v(x) \right\}, \tag{5.2}$$

for some non-negative weights (q_v) that sum to 1. We classically assume that node v in the graph has access to unbiased stochastic gradients of f_v (of the form $F_v(x, \xi_v)$). The standard goal of decentralized optimization is to minimize f using only local computations and communications (only neighboring nodes in the graph can communicate).

Notations. Standard small letters $(x, g, y, z,$ etc) are for vectors in $\mathbb{R}^d$. Capital letters (mostly W) are for matrices in $\mathbb{R}^{\mathcal{V}\times\mathcal{V}}$. Bold letters $\mathbf{x}, \mathbf{g}, \dots$ are for *concatenated vectors* in $\mathbb{R}^{\mathcal{V}\times d}$, that we write as $\mathbf{x} = (x_v)_{v\in\mathcal{V}}$. For some vector $x \in \mathbb{R}^d$, we denote $\mathbf{x} \in \mathbb{R}^{\mathcal{V}\times d}$ the concatenated vector such that $\mathbf{x}_v = x$ for all $v \in \mathcal{V}$. $\mathbb{1} \in \mathbb{R}^{\mathcal{V}}$ is the vector with all entries equal to 1. For $\mathbf{x} \in \mathbb{R}^{\mathcal{V}\times d}$, we denote $\bar{\mathbf{x}} = \frac{1}{n}\mathbb{1}\mathbb{1}^\top\mathbf{x}$, where $n = |\mathcal{V}|$.

In this chapter we study a general scheme for **asychronous SGD on graphs (AGRAF)** which is summarized in Algorithm 5.1: workers asynchronously perform local SGD steps (lines 3-4), while an underlying **linear communication algorithm** is running *without incurring communication delays* (line 7). A linear communication algorithm on graph G implies that any communication update can be formulated as $\mathbf{x}_+ = W\mathbf{x}_-$ where $\mathbf{x}_+, \mathbf{x}_- \in \mathbb{R}^{\mathcal{V}\times d}$ are respectively the global state after and before the communication update, and $W \in \mathbb{R}^{\mathcal{V}\times\mathcal{V}}$ is a **communication matrix** with $W_{v,w}$ being zero for disconnected nodes v, w, i.e. $W_{v,w} \neq 0$ iff $\{v, w\} \in \mathcal{E}$.

Since every agent asynchronously works at their own speed and communicates in a decentralized way, there is no global state. Keeping track of a global ordering of the iterates involving both computation and communication updates is thus a challenge. In the next

[1] Since we consider varying topologies, this graph should be thought as the union of graphs considered over time.

Algorithm 5.1: Asynchronous SGD on graph G (AGRAF SGD)

Input: $\bar{x}^0 \in \mathbb{R}^d$, $x_v = \bar{x}^0$ for $v \in \mathcal{V}$ initialized local variables, stepsize $\gamma > 0$
for $v \in \mathcal{V}$, **do**
  Upon finishing computation of a stochastic gradient $\nabla F(\tilde{x}_v, \tilde{\xi}_v)$ at some previous local current state $\tilde{x}_v$,

$$x_v \longleftarrow x_v - \gamma \nabla F_v(\tilde{x}_v, \tilde{\xi}_v)\,.$$

  Compute $\nabla F_v(x_v, \xi_v)$ for $\xi_v \sim \mathcal{D}_v$ independently from the past, at current state x_v.
end for
while procedure still running **do**
  Run any **linear communication algorithm** on graph G incurring no communication delay.
end while

subsection we address this challenge and propose a way to effectively cast Algorithm 5.1 into equations with ordered updates. This reformulation is a key novelty. It allows for an improved theoretical analysis with better rates together with relaxed communication and computations assumptions, allowing AGRAF SGD to cover asynchronous versions of many popular distributed and federated algorithms.

5.2.2. The sequence studied

We denote by $T_0 = 0$ the initialization time of the algorithm and by $\{0 < T_1 < T_2 < \ldots\}$ the times at which the local computation updates are made. Note that these are physical (continuous) times, and that several agents may possibly finish their local computations at the same time T_k. We also assume that computational updates are **atomic**. For some time T, we denote as $T-$ the left limit ($\lim_{t \to T, t<T}$) and $T+$ the right limit ($\lim_{t \to T, t>T}$). For time $t \in \mathbb{R}^+$ (physical time), let $x_v(t) \in \mathbb{R}^d$ denote the state of the local variable at time t, and let $\mathbf{x}(t) = (x_v(t))_{v \in \mathcal{V}}$. For $k \geq 0$ and $v \in \mathcal{V}$, let x_v^k denote the state of the local variable at node v at time T_k+ *i.e.*, $x_v^k = x_v(T_k+) = \lim_{t \to T_k, t>T_k} x_v(t)$ and let $\mathbf{x}^k = (x_v^k)_{v \in \mathcal{V}}$.

Communication updates. For $k \geq 0$, none to plenty of communication updates may have happened between the computational update times T_k and T_{k+1}. We encode these communication updates by a *single* matrix W_k: W_k is thus the product of all communication matrices corresponding to communication updates between times T_k and T_{k+1}. Hence, we can write:

$$\mathbf{x}(T_{k+1}-) = W_k \mathbf{x}(T_k+)\,.$$

If no communication happened between two gradients computed, we have $W_k = I_d$. If there are r communications between times T_k+ and $T_{k+1}-$ that happened at times $T_k < T_{k,1} < \ldots < T_{k,r} < T_{k+1}$, denoting by $W_{k,r}$ the communication matrix corresponding to communication updates at time $T_{k,r}$, we have $W_k = W_{k,r} \cdot \ldots \cdot W_{k,2} \cdot W_{k,1}$. Note that for $r = 0$ this product is taken equal to I_d.

Computation updates. For $k \geq 1$, let $\mathcal{I}_k \subset \mathcal{V}$ be the set of nodes that finish computing stochastic gradients $\nabla F_v(\tilde{x}_v^k, \tilde{\xi}_v^k)$ for $v \in \mathcal{I}_k$ at time T_k-. The computation updates, that are assumed to be **atomic**, then read:

$$x_v(T_k+) = x_v(T_k-) - \gamma \nabla F(\tilde{x}_v^k, \tilde{\xi}_v^k)\,, \quad v \in \mathcal{I}_k\,,$$

where $\tilde{x}_v^k = x_v^{k-1-\tau(k,v)}$ and $\tilde{\xi}_{v_k}^k = \xi_{v_k}^{k-1-\tau(k,v)}$, for $\tau(k,v) \geq 0$ the delay of this update that corresponds to the number of computation updates performed by other nodes during the computation of the local stochastic gradient.

The sequence studied. Combining communication and computation updates, the sequence generated by Algorithm 5.1 follows the following recursion:

$$\mathbf{x}^{k+1} = W_k \mathbf{x}^k - \gamma \mathbf{g}^k \,, \tag{5.3}$$

where $g_w^k = 0$ for $v \notin \mathcal{I}_k$, and $g_v^k = \nabla F_v(x_v^{k-\tau(k,v)}, \xi_v^{k-\tau(k,v)})$.

What is important to keep in mind is that the iterates $\mathbf{x}^{k+1}$ are taken at the time just after computation updates (time T_k+) so that k denotes the number of computation updates. $\mathcal{I}_k$ is the set of nodes that perform computation updates at iteration k, it can be any subset of $\mathcal{V}$, and $\sum_{k<K} |\mathcal{I}_k|$ denotes the total number of stochastic gradients computed up to iteration K by **all** the agents. The matrix W_k encodes all communications that happened between the k-th and $(k+1)$-th computation updates (there can be any number such communications, the more there are the more (W_k) will mix).

5.2.3. AGRAF SGD is the right formulation of Asynchronous Decentralized SGD

Recall that the Decentralized SGD algorithm [KSJ22, e.g.] consists in iterations of the form:

$$x_v^{k+1} = \sum_{w \sim v} W_{\{v,w\}}^{(k)} x_w^k - \gamma \nabla F_v(x_v^k, \xi_v^k) \,, \quad \forall v \in \mathcal{V} \,, \tag{5.4}$$

for communication matrices $(W^{(k)})_k$ satisfying Assumption 5.3.2. The question thus arises: *how can Decentralized SGD be turned into an asynchronous algorithm?* Previous works [LHLL15, BRW+23] proposed and analyzed schemes that take the following form: at each iteration, **one** node v_k is sampled independently from the past (with fixed or lower bounded probability), and this node performs a local stochastic gradient step together with an averaging operation with its neighbors in the graph. This results in updates of the form of AGRAF SGD, for $\mathcal{I}_k = \{v_k\}$ and W_k a matrix that depends on v_k and that mixes (in mean) independently from the past ($\mathbb{E}[W_k | W_0, \dots, W_{k-1}]$ mixes well).

Leaving the analyses aside, this prior approach is too restrictive: **(i)** *communication assumptions* do not allow varying topologies that may mix but only in the long run, which may particularly be the case for asynchronous algorithms, and **(ii)** *computation assumptions* do not allow for more than one worker to update their value at the same time; having a sampling assumption restricts the type of delays that the algorithm can handle; and nodes that compute should not necessarily be correlated to communicating edges since this forbids the use of several local SGD steps.

AGRAF SGD thus appears as a natural way to make Decentralized SGD asynchronous: nodes are not forced to all perform computations at the same time as in eq. (5.4), and having the relaxed communication assumption (Assumption 5.3.2) allows any communication order, especially when one considers W_k as a concatenation of all communications that may happen between two consecutive computations.

5.2.4. Some examples covered by AGRAF

We now give a few examples of algorithms (*i.e.* communication and computation schedules) that can be cast as AGRAF SGD. The three first are degenerate cases.

Minibatch SGD and **Asynchronous SGD** are obtained by setting $W_k = \frac{1}{n}\mathbb{1}\mathbb{1}^\top$, and $\mathcal{I}_k = \mathcal{V}$ and $\mathcal{I}_k = \{v_k\}$ for some node v_k respectively.

Decentralized (local) SGD. Set $\mathcal{I}_k = \mathcal{V}$ and $(W_k)_k$ a sequence of gossip matrices to obtain Decentralized SGD [RNV10]. Note that in that case there are no computation delays, since this algorithm is inherently synchronous and all nodes perform updates at the same time. As done in [KLB+20], periodic communications are possible, allowing to recover algorithms with several local gradient steps between each communication round, such as **Local (Decentralized) SGD** [Sti18] or **FedAvg** [MMR+17].

Asynchronous Decentralized SGD. As explained in Section 5.2.3, AGRAF SGD covers Asynchronous Decentralized SGD beyond particular instances previously studied [LHLL15, BRW+23]. Furthermore, since we make relaxed communication/computation assumptions, we cover more general decentralized algorithms that allow **local** gradient steps between communications, varying topologies, and arbitrary computations. As such, together with covering an asynchronous version of Decentralized SGD (5.4), we also cover asynchronous versions of **FedAvg** or**Local SGD**, together with **FedBuff** [NMZ+22].

Asynchronous SGD on Loss Networks. If communication latencies are not negligible compared to computational ones, designing an algorithm that is asynchronous and decentralized becomes much more challenging, as the naive implementation might lead to deadlocks. In order to handle non-negligible communication delays, we use **loss networks** [Kel91b] to enforce that the edges adjacent to *"busy"* nodes are prohibited to be used for communicating[2]. This enables us to design communication/computation schemes that fit in the AGRAF framework, while not violating the physical delay constrains. We introduce these Loss Networks in Section 5.5.2: we define them more thoroughly, and provide their ergodic mixing properties with explicit constants that depend on the graph topology and local communication and computation delays.

5.3. Assumptions and Notations

We consider solving the problem (5.2) under several standard [Bub15, see, e.g.,] combinations of conditions on the objective F. We denote the minimum of f as $f^* = \min_{x\in\mathbb{R}^d} f(x)$, an upper bound on the **initial suboptimality** $\Delta \geq f(\bar{x}^0) - f^*$, and an upper bound on the **initial distance** to the minimizer $D \geq \min\{\|x_0 - x^*\| : x^* \in \arg\min_x f(x)\}$ that we assume to exist. $\|\cdot\|$ denotes the Euclidean norm. When F_v and f_v are convex, we do not necessarily assume they are differentiable, but we abuse notation and use $\nabla f_v(x)$ and $\nabla F_v(x;\xi)$ to denote an arbitrary subgradient at x. The loss F_v is B-**Lipschitz-continuous** if for each x, y and ξ, we have $|F_v(x;\xi) - F_v(y;\xi)| \leq B\|x - y\|$. The objective f_v is L-**smooth** if it is differentiable and its gradient is L-Lipschitz-continuous. We assume the stochastic gradients have σ^2-**bounded variance**[3].

Assumption 5.3.1 (Noise)**.** *There exists σ^2 such that for all x and $v \in \mathcal{V}$, we have $\mathbb{E}\left[\nabla_x F_v(x,\xi_v)\right] = \nabla f_v(x)$ and $\mathbb{E}\left[\|\nabla f_v(x) - \nabla_x F_v(x,\xi_v)\|^2\right] \leq \sigma^2$, where $\xi_v \sim \mathcal{D}_v$.*

5.3.1. Graph, communications and mixing.

We now formulate the communication assumptions we will make. For $k \geq 0$, as opposed to some previous Asynchronous Decentralized SGD analyses [LHLL15, BRW+23], we do not want to assume that W_k mixes well in mean (i.e., that the spectral gap of $\mathbb{E}\left[W_k | W_0, \dots, W_{k-1}\right]$ is non-null or some other related assumption), since W_k may possibly be the identity matrix.

[2] Loss Networks were initially introduced by F. Kelly to model telecommunication networks, where the same mobile phone cannot initiate another phone call while being busy with another call. In our case, phone calls should be thought as communicating with a neighbor.

[3] which can easily be generalized to $\mathbb{E}\left[\|\nabla f_v(x) - \nabla_x F_v(x,\xi_v)\|^2\right] \leq \sigma^2 + \delta^2\|\nabla f_v(x)\|^2$.

We use the least restrictive assumption under which convergence of (synchronous) decentralized SGD is established [KLB+20], by assuming that if we wait enough communication updates, a consensus will ultimately be achieved.

Assumption 5.3.2 (Ergodic mixing). *$W_k \in [0,1]^{\mathcal{V}\times\mathcal{V}}$ is symmetric, $W_k\mathbb{1} = \mathbb{1}$, and there exist $\rho, k_\rho > 0$ such that we have $\forall k \in \mathbb{N}$ and $\forall \mathbf{x} \in \mathbb{R}^{\mathcal{V}}$:*

$$\mathbb{E}\left[\left\|W^{(k:k+k_\rho)}\mathbf{x} - \bar{\mathbf{x}}\right\|^2 | \mathcal{F}_k\right] \leq (1-\rho)^2\|\mathbf{x} - \bar{\mathbf{x}}\|^2, \tag{5.5}$$

where for $k, \ell \geq 0$, $W^{(k,\ell)} = W_{\ell-1}\dots W_{k+1}W_k$, and $\mathcal{F}_k = \sigma(\mathbf{x}^s, \mathbf{g}^{s-1}, W_{s-1}, s \leq k)$ is the filtration up to step k.

This assumption makes it possible to consider any "reasonable" communication scheme. In the rest of the chapter, when assuming that Assumption 5.3.2 holds for some constants (ρ, k_ρ), we write $\bar{\rho} = \frac{e-1}{e}\frac{\rho}{k_\rho}$ (with $e = \exp(1)$), and this quantity is used in our main results.

5.3.2. Heterogeneous and homogeneous settings, sampling assumptions.

Assuming that the sequence of nodes $(\mathcal{I}_k)_{k\geq 0}$ that iteratively perform local updates is arbitrary makes it possible to encompass all possible computation orderings and cover arbitrary delays. It is much more general than assuming that $\mathcal{I}_k = \mathcal{V}$ for all k (decentralized SGD) or $\mathcal{I}_k = \{v_k\}$ for v_k sampled independently from the past, as assumed in most previous asynchronous decentralized works [LHLL15, BRW+23]. However, if functions f_v are not all equal and if the sequence v_k is arbitrary, convergence to the global function f cannot be assured (some of the nodes v might simply never appear during training). We therefore need to make some sampling assumption if we assume that local functions can be heterogeneous. We will thus assume either one the two following assuptions: **(i)** the heterogeneous setting where local functions f_v can be different, but where we make some node-sampling assumption for computations, and **(ii)** the homogeneous setting, where computations can be arbitrary, but functions f_v are all the same. Note that it is classical in asynchronous optimization to either assume **(i)** or **(ii)**; for instance, Asynchronous SGD with arbitrary orderings is proved to converge only under such assumptions [MBEW22, KSJ22]. However, Asynchronous Decentralized works only assume that the sampling assumption **(i)** holds. Formally, we summarize these into the following two assumptions.

Assumption 5.3.3 (Heterogeneous setting). *There exists ζ^2 such that the* **population variance** *satisfies:*

$$\sum_{v\in\mathcal{V}} q_v\|\nabla f_v(x) - \nabla f(x)\|^2 \leq \zeta^2\,, \quad \forall x \in \mathbb{R}^d\,. \tag{5.6}$$

There exists $\mathbf{p} = (p_v)_{v\in\mathcal{V}} \in [0,1]^{\mathcal{V}}$ such that the sequence $(\mathbb{1}_{v\in\mathcal{I}_k})_{k\geq 0}$ is i.i.d. *distributed, with $\mathbb{P}(v \in \mathcal{I}_k) = p_v$ for all $k \geq 0, v \in \mathcal{V}$. We denote $\kappa_{\mathbf{p}} = \frac{p_{\max}}{\bar{p}}$, $p_{\max} = \max_v p_v$ and $\bar{p} = 1/n\sum_{v\in\mathcal{V}} p_v$ Furthermore, we assume that $\mathbf{p}$ is proportional to $\mathbf{q}$: $\mathbf{p} = \beta\mathbf{q}$, and since $\sum_v q_v = 1$, we thus have $\beta = n\bar{p}$.*

Assumption 5.3.4 (Homogeneous setting). *All functions f_v satisfy $f_v \equiv f$. No assumption on $(\mathcal{I}_k)_{k\geq 0}$.*

5.4. General Convergence Analysis

We now turn to our main results: convergence guarantees for AGRAF SGD, under a variety of regularity assumptions and settings. Note that in almost all cases, our rates do not depend on any upper bound on the maximal delays, which is a key feature of our analysis. This is also the case for asynchronous SGD [KSJ22, MBEW22] or a recent asynchronous decentralized SGD work [BRW+23]. In this section, while presenting the results, we will only

compare our results to degenerate baselines such as minibatch SGD, asynchronous SGD or decentralized SGD, in order to give simple arguments to show that our rates have expected order of magnitudes, leaving more complex comparisons and applications to be developed in Section 5.5. We first start with convex-Lipschitz losses. In this section, all the rates are obtained for **a constant stepsize** γ, explicited in the proofs in the Appendix.

Theorem 5.1 (Lipschitz-convex rate)**.** *Assume that f is convex and that for almost all (i.e., with probability 1) $\xi \sim \mathcal{D}_v$ $F_v(\cdot, \xi)$ is B-Lipschitz for some $B > 0$, let $D^2 \geq \|x_0 - x^\star\|^2$, and $F_K = \mathbb{E}\left[f\left(\frac{1}{\sum_{k<K}|\mathcal{I}_k|}\sum_{k=0}^{K-1}\sum_{v\in\mathcal{I}_k} x_v^k\right) - f(x^\star)\right]$.*

1. *In the* **homogeneous** *setting (Assumption 5.3.4),*

$$F_K = \mathcal{O}\left(\sqrt{\frac{B^2D^2n\bar{\rho}^{-1}}{\sum_{k<K}|\mathcal{I}_k|}}\right).$$

2. *In the* **heterogeneous** *setting (Assumption 5.3.3),*

$$F_K = \mathcal{O}\left(\sqrt{\frac{B^2D^2}{\sum_{k<K}|\mathcal{I}_k|} \times n\sqrt{p_{\max}}(\sqrt{\kappa_{\mathbf{p}}} + \bar{\rho}^{-1})}\right).$$

We thus recover the well-known rate of minibatch SGD for convex-Lipschitz losses, for $\bar{\rho} = \Theta(1)$ and $|\mathcal{I}_k| = n$, leading to the optimal rate $\mathcal{O}(\sqrt{B^2D^2/K})$ [NY83]. Asynchronous SGD has also been studied under such assumptions, with the rate $\mathcal{O}(\sqrt{B^2D^2n/K})$ that we recover here ($\bar{\rho} = 1$ and $|\mathcal{I}_k| = 1$) [MBEW22], that is minmax optimal [WWS+18]. No rates under the given assumptions existed for Decentralized (local) SGD, that thus exhibits a rate of $\mathcal{O}(\sqrt{B^2D^2\bar{\rho}^{-1}/K})$. Finally, adding the sampling assumption not only enables to handle heterogeneous functions, but also leads to improved rates: for well balanced weights ($p_v \approx \bar{p}$ and $\kappa_{\mathbf{p}} \approx 1$) we have $n\sqrt{\bar{p}}\bar{\rho}^{-1}$ instead of $n\bar{\rho}^{-1}$, which can improve the rate by a factor $1/\sqrt{n}$ if $\mathcal{O}(1)$ agents compute at the same time, which is usually the case in the asynchronous setting. This phenomenon (better rates under the sampling assumption) appears in all our other rates below.

Theorem 5.2 (Lipschitz-smooth-convex rate)**.** *Assume that f is convex, for almost all $\xi \sim \mathcal{D}$, $F(\cdot, \xi)$ is B-Lipschitz for some $B > 0$, f_v is L-smooth, Assumption 5.3.1 holds, and let $D^2 \geq \|x_0 - x^\star\|^2$. In the* **homogeneous** *setting,*

$$\begin{aligned}\mathbb{E}\left[f\left(\frac{1}{\sum_{k<K}|\mathcal{I}_k|}\sum_{k=0}^{K-1}\sum_{v\in\mathcal{I}_k} x_v^k\right) - f(x^\star)\right] \\ = \mathcal{O}\left(\frac{L\bar{\rho}^{-1}nD^2}{\sum_{k<K}|\mathcal{I}_k|}\sqrt{\frac{\sigma^2B^2}{\sum_{k<K}|\mathcal{I}_k|}}\right. \\ \left. + \left(\frac{D^2n\sqrt{L\left(B^2 + \bar{\rho}^{-1}\sigma^2\right)}}{\sum_{k<K}|\mathcal{I}_k|}\right)^{\frac{2}{3}}\right)\end{aligned}$$

For Lipschitz-smooth functions, setting $\bar{\rho}^{-1} = 1$ and $|\mathcal{I}_k| = 1$, we recover the exact same rates as Asynchronous SGD under arbitrary delays, recently derived by [MBEW22, KSJ22], and that do not depend on any upper bound on the delays. These rates are thus extended to the more general AGRAF SGD algorithm.

Theorem 5.3 (Smooth-convex)**.** *Assume that f is convex, all f_v are L-smooth, and let $D^2 \geq \|x_0 - x^\star\|^2$.*

1. *In the* **homogeneous** *setting,*

$$\mathbb{E}\left[f\left(\frac{1}{\sum_{k<K}|\mathcal{I}_k|}\sum_{k=0}^{K-1}\sum_{v\in\mathcal{I}_k}x_v^k\right)-f(x^\star)\right]$$
$$=\mathcal{O}\left(\frac{LD^2(n\bar{\rho}^{-1}+\sqrt{n\tau_{\max}})}{\sum_{k<K}|\mathcal{I}_k|}+\sqrt{\frac{D\sigma^2}{\sum_{k<K}|\mathcal{I}_k|}}\right.$$
$$\left.+\left[\frac{D^2\sqrt{L\sigma^2n^2\bar{\rho}^{-1}}}{\sum_{k<K}|\mathcal{I}_k|}\right]^{2/3}\right),$$

where $\tau_{\max}\geq\sup_{k<K,v\in\mathcal{V}}\sum_{\ell=k}^{\tau(k-1,v)}|\mathcal{I}_\ell|$ *is an upper bound on the maximal compute delay.*

2. *In the* **heterogeneous** *setting,*

$$\mathbb{E}\left[f\left(\frac{1}{K}\sum_{k<K}\bar{x}^k\right)-f(x^\star)\right]$$
$$=\mathcal{O}\left(\frac{LD^2\sqrt{\kappa_{\mathbf{P}}}\left(\frac{1}{\bar{p}}+(\bar{\rho}\sqrt{\bar{p}})^{-1}\right)}{K}+\sqrt{\frac{D^2(\sigma^2+\zeta^2)}{n\bar{p}K}}\right.$$
$$\left.+\left[\frac{D^2\sqrt{L\sigma^2p_{\max}\bar{\rho}^{-1}+L\zeta p_{\max}\bar{\rho}^{-2}}}{\bar{p}K}\right]^{\frac{2}{3}}\right).$$

Removing the Lipschitz assumption, we are still able to recover and extend the rates of Asynchronous SGD with **constant** stepsizes. Note that under no sampling assumption, this rate depends on $\sqrt{n\tau_{\max}}$ instead of n as in the previous two theorems; however, this dependency is still better than depending on $\tau_{\max}$ since we always have $\tau_{\max}\geq n$. We expect to be able to remove this dependency by the use of varying stepsizes as was done for Asynchronous SGD (where stepsizes scale as $1/(L\tau(k))$, inversely proportional to the actual delay). However, such stepsizes cannot be used in a fully decentralized setting, since a given node cannot be aware of the iteration counter k and thus of the delay $\tau(k)$. Note also that in the sampling case, we have $\mathbb{E}\left[\sum_{k<K}|\mathcal{I}_k|\right]=n\bar{p}K$, so that the statistical rate is still reached. These comments also applies to the **non-convex and smooth setting** below, for which we fall back to showing that the algorithm will find an approximate first-order stationary point of the objective. We recover, as in the convex-smooth case just above, the exact same rates as [KLB+20] for Decentralized (local) SGD.

Theorem 5.4 (Non-convex and smooth rates)**.** *Assume that the functions* f_v *are* L*-smooth.*

1. *In the* **homogeneous** *setting,*

$$\mathbb{E}\left[\frac{1}{\sum_{k<K}|\mathcal{I}_k|}\sum_{k<K}|\mathcal{I}_k|\left\|\nabla f(\bar{x}^k)\right\|^2\right]$$
$$=\mathcal{O}\left(\frac{LF_0(\sqrt{n\tau_{\max}}+n\bar{\rho}^{-1})}{K}+\left(\frac{L\sigma^2F_0}{K}\right)^{\frac{1}{2}}\right.$$
$$\left.+\left(\frac{L\sigma nF_0}{K\sqrt{\bar{\rho}}}\right)^{\frac{2}{3}}\right).$$

Table 5.1 – *We compare the number of iterations required to reach the statistical regime* $\mathcal{O}(\sigma^2/\sum_{k<K}|\mathcal{I}_k|)$ *(if it is reached) of previous Asynchronous Decentralized SGD works [LHLL15, BRW+23] with our rates.* **Strong communication assumption** *: Assumption 5.3.2 with* $k_\rho = 1$ *and* W_k *independent from the past;* **Sampling assumption** *:* $\mathcal{I}_k = \{v_k\}$ *with* v_k *i.i.d. sampled or* $\mathbb{P}(v \in \mathcal{I}_k) = p_v$ *i.i.d. sampled.* [(a)] *[BRW+23] reaches* $\mathcal{O}(\bar{\rho}^{-2}\sqrt{\sigma^2/K})$ *instead, after* $\mathcal{O}(n^2\bar{\rho}^{-4})$ *iterations.*

Reference	Communication Assumption	Computation Assumption	Regularity	# iterations before statistical regime
[LHLL15]	**Strong**	Sampling	Smoothness	$\mathcal{O}(\max(n^4\bar{\rho}^{-4}, \tau_{\max}^4))$
[BRW+23]	**Strong**	Sampling	Smoothness	N.A.[(a)]
Theorem 5.2.1 (convex)	Assumption 5.3.2	**Arbitrary**	Smooth-**Lipschitz**, **Homogeneous**	$\mathcal{O}(n^4\bar{\rho}^{-2})$
Theorem 5.4.1	Assumption 5.3.2	**Arbitrary**	Smooth, **Homogeneous**	$\mathcal{O}(\max(n^4\bar{\rho}^{-2}, n\tau_{\max}))$
Theorem 5.4.2	Assumption 5.3.2	Sampling	Smoothness	$\mathcal{O}(n^2\bar{\rho}^{-4})$

2. *In the* **heterogeneous** *setting,*

$$\mathbb{E}\left[\frac{1}{K}\sum_{k<K}\left\|\nabla f(\bar{x}^k)\right\|^2\right] = \mathcal{O}\left(\frac{LF_0\sqrt{\kappa_{\mathbf{p}}}(\frac{1}{\bar{p}} + (\bar{\rho}\sqrt{\bar{p}})^{-1})}{K} + \left(\frac{L(\sigma^2+\zeta^2)F_0}{K}\right)^{\frac{1}{2}} + \left(\frac{LnF_0\sqrt{\sigma^2 p_{\max}\bar{\rho}^{-1} + \zeta^2 p_{\max}\bar{\rho}^{-2}}}{K}\right)^{\frac{2}{3}}\right).$$

Remark 5.4.1 (Heterogeneous without sampling)**.** *So far, the heterogeneous setting was only considered under a sampling assumption. In fact, generalizing [MBEW22, Theorem 4] to AGRAF SGD, under both* **heterogeneous functions** *with population variance* ζ^2 *(as in eq. (5.6)) and* **arbitrary ordering of the updates***, the exact same rate as Theorem 5.4.1 up to an additional term* $\mathcal{O}(\zeta^2)$ *could be obtained.*

5.5. Applications

5.5.1. Better rates for Asynchronous Decentralized SGD

A first direct application of our theory is a better analysis of Asynchronous Decentralized SGD. Comparing our analysis with those of [LHLL15, BRW+23], we highlight that our work handles arbitrary computation orders and delays in the homogeneous settings, as opposed to [LHLL15, BRW+23] that are only valid for $k_\rho = 1$ in Assumption 5.3.2 (which means that at any step k, conditionally on the current state, the graph of edges that can be sampled must be connected) and under a sampling assumption. In both homogeneous and heterogeneous cases, our communication assumptions are much less restrictive. Furthermore, under similar computation and regularity assumptions as [LHLL15, BRW+23] (sampling and smooth losses, see last line of Table 5.1), our convergence bound (Theorem 5.3.2) reaches a statistical rate $\sqrt{\sigma^2/\sum_{k<K}|\mathcal{I}_k|}$ after $\sum_{k<K}|\mathcal{I}_k| = \mathcal{O}(n^2\bar{\rho}^{-4})$, while [BRW+23] does not reach such a statistical rate and [LHLL15] reaches this rate after $K = \mathcal{O}(\max(n^4\bar{\rho}^{-4}, \tau_{\max}^4))$ iterations. For the sake of comparison, we take $\bar{p}$ of order 1 in our rates.

5.5.2. Asynchronous Decentralized SGD on Loss Networks

The previous considerations and the AGRAF SGD rates hold as long as there is no communication delay. The following question then arises: given a communication graph $G = (\mathcal{V}, \mathcal{E})$ with communication delays $\tau_{\{v,w\}}$ and computation delays τ_v for $v \in \mathcal{V}$ and

$\{v, w\} \in \mathcal{E}$, can we reverse-engineer and build communication/computation schemes that fit in the AGRAF SGD framework and that do not break the communication and computation constraints? Can we analyze such a scheme and prove that it mixes well (in the sense that Assumption 5.3.2 holds, for explicit values of ρ, k_ρ) ?

Overview of the Loss Network scheme. Starting with $\mathcal{I}_k = \{v_k, w_k\} \in \mathcal{E}$, and communication matrices W_k corresponding to an averaging along the edge $\{v_k, w_k\}$ as a baseline (*i.e.*, $W_k = I_{\mathcal{V}} - \frac{(e_{v_k} - e_{w_k})(e_{v_k} - e_{w_k})^\top}{2}$ where (e_v) is the canonical basis of $\mathbb{R}^{\mathcal{V}}$) as a baseline, choosing a sequence $\mathcal{I}_k$ such that there is no induced communication delays becomes tricky. While assuming that $\{v_k, w_k\}$ is sampled independently from the past with fixed probability [LHLL15] is amenable for the analysis (since then Assumption 5.3.2 directly holds for $k_\rho = 1$), this can incur communication delays if for instance the same node is sampled in two consecutive updates.

To alleviate this issue, we remove the independence between sampled edges in the following way: we impose that nodes that are already involved in a communication are tagged as *busy*, and that busy nodes cannot be involved in new communications. Then once a node finishes a computation, it can then choose a new neighbor (*who is not busy*) to start communicating with. Doing so, the induced communication matrices are no longer independent, as they follow a Markov process. This scheme is inspired by **Loss-Networks**, introduced in [Kel91b] to model telecommunication networks, in which an edge in the graph models a phone communication that can happen; since a phone cannot make several calls in parallel, once involved in a communication with some neighboring node it cannot be called by another neighbor while it is busy; this is exactly the same process we use, phone calls being replaced by model communications.

[BGPS06] consider a model (without any delay) for gossip algorithms, where updates are that of Equation (5.15) without the gradient steps, and these updates happen at the times of Poisson point processes (a *P.p.p.* of intensity $p_{\{v,w\}}$ for an update along $\{v, w\}$). Consequently, W_k is independent from the past, and $\mathbb{P}\left(W_k = W_{\{v,w\}}\right) \propto p_{\{v,w\}}$.

The *P.p.p. model* considered in [BGPS06] where the updates are performed at the times of *Poisson point processes* is particularly amenable to analysis, but it assumes that communications and computations are done instantaneously. Thus, actual implementations differ from its underlying assumptions, unless further synchrony is assumed. To alleviate this issue, with pairwise communications ruled by point processes as a baseline, we consider a protocol in which nodes are tagged as *busy* when they are already engaged in an update, and communications between busy nodes are forbidden. Our model is inspired from classical Loss Network models [Kel91b], in which edges are activated following the same procedure as in the *P.p.p. model*, with a *P.p.p.* of intensity $p_{\{v,w\}}$. Note that we do not consider these intensities to be constraints of the problem, but rather parameters of the algorithm, that can be tuned. Each node has an exponential clock of intensity $p_v = \frac{1}{2} = \sum_{w \sim v} p_{\{v,w\}}$. At each clock-ticking, if v is not busy, it selects a neighbor w with probability $p_{\{v,w\}} / \sum_{u \sim v} p_{\{u,v\}}$. If w is not *busy*, v and w compute and exchange information, becoming busy for a duration $\tau'_{\{v,w\}}$. We can think of this procedure as classical gossip on an underlying random graph that follows a Markov-Chain process. The difference between our communication model on Loss Networks and the P.p.p. model lies in that in our case, W_k is not independent on the past. In fact, we have:

$$\mathbb{P}\left(\{v_k, w_k\} = \{v, w\} | \mathcal{F}_k\right) = \frac{\mathbb{1}_{\{v,w \text{ not busy at time } T_k\}} p_{\{v,w\}}}{\sum_{\{u,u'\} \in \mathcal{E}} \mathbb{1}_{\{u,u' \text{ not busy at time } T_k\}} p_{\{u,u'\}}},$$

leading to complicated intricacies between the matrices $(W_k)_k$, that we need to handle.

Proving Theorem 5.5 requires to show that there exist ρ, k_ρ (that need to be computed)

such that for any $k \geq 0$, $\mathbf{x} \in \mathbb{R}^{\mathcal{V}}$,

$$\mathbb{E}\left[\left\|W_{\{v_{k+k_\rho-1},w_{k+k_\rho-1}\}} \cdot \ldots \cdot W_{\{v_k,w_k\}}(\mathbf{x}-\bar{\mathbf{x}})\right\|^2 |\mathcal{F}_k\right] \leq (1-\rho)^2\|\mathbf{x}-\bar{\mathbf{x}}\|^2 .$$

How to schedule such a process ? If nodes start a new communication right after they finish their last one, the process can end up in deadlock and thus does not mix at all: this is for instance the case on the cycle or line graphs with an even number of nodes [Kel91b]. We thus need to introduce some randomness and some waiting times. We proceed as follows and use exponential random waiting times as in [Kel91b].

(i) Once a node v finishes a communication, it waits a time $T_v \sim \text{Exp}(p_v)$ (exponential random variable, of intensity p_v).

(ii) If v is still not busy after this waiting time, v samples some neighboring node $w \sim v$ with probability $\frac{p_{\{v,w\}}}{p_v}$ to communicate with, for $\sum_{w\sim v} p_{\{v,w\}} = p_v$.

(iii) If w is busy, this procedure restarts at **(i)**, else both v and w become busy and can communicate. Once they are busy, they cannot communicate with other nodes. The communication between v and w consists in averaging local values by setting x_v, x_w to $(x_v + x_w)/2$. When this is done, they each perform a local (eventually delayed) gradient step, and then become *non-busy*. Overall, the k^{th} update reads:

$$x_{v_k}^{k+1} = \frac{x_{v_k}^k + x_{w_k}^k}{2} - \gamma \nabla F_{v_k}\left(x_{v_k}^{k-\tau(v_k,k)}, \xi_{v_k}^{k-\tau(v_k,k)}\right), \tag{5.7}$$

and similarly at node w_k. The procedure described just above (**(i)-(ii)-(iii)**) to sample pairs of nodes that iteratively perform computations and pairwise communications can be instantiated locally, provided nodes know when their neighbors in the graph are busy — this can be relaxed by adding some "busy-checking" operation. However, the key challenge here lies in that the communication matrices $(W_k)_{k\geq 0}$ induced by the updates Equation (5.7) are not independent, and analyzing some form of ergodic mixing time becomes highly non-trivial. Still, using the randomness introduced in this procedure through the exponential waiting times and the sampling of neighbors, we are able to prove that Assumption 5.3.2 holds, for values of ρ, k_ρ that depend on the physical delays.

Assumption 5.5.1 (Loss Network assumptions)**.** *There exist* $\tau_v, \tau_{\{v,w\}} \in \mathbb{R}_{>0}$[4] *for* $v \in \mathcal{V}, \{v,w\} \in \mathcal{E} > 0$ *such that a communication between* v *and* w *takes a time at most* $\tau_{\{v,w\}}$*, and computing a stochastic gradient at node* v *takes a time at most* τ_v*.*

The updates of *decentralized SGD on loss networks* write as:

$$\begin{cases} x_{v_k}^{k+1} = \dfrac{x_{v_k}^k + x_{w_k}^k}{2} - \gamma \nabla F_{v_k}\left(x_{v_k}^{k-\tau(v_k,k)}, \xi_{v_k}^{k-\tau(v_k,k)}\right) \\ x_{w_k}^{k+1} = \dfrac{x_{v_k}^k + x_{w_k}^k}{2} - \gamma \nabla F_{w_k}\left(x_{w_k}^{k-\tau(w_k,k)}, \xi_{w_k}^{k-\tau(w_k,k)}\right) \end{cases}, \tag{5.8}$$

leading to $\mathbf{x}^{k+1} = W_k\mathbf{x}^k - \gamma\mathbf{g}^k$, for $W_k = W_{\{v_k,w_k\}} = I_{\mathcal{V}} - \frac{(e_{v_k}-e_{w-k})(e_{v_k}-e_{w-k})^\top}{2}$, and $\mathbf{g}^k$ the corresponding delayed gradients. Note then that this takes the same form as the **AGRAF SGD** sequence.

Theorem 5.5. *Under Assumption 5.5.1, assume that*

$$p_{\{v,w\}} = \min\left(\frac{1}{\max_{u\sim v}\tau'_{\{u,w\}}}, \frac{1}{2(\max(d_v,d_w)-1)\tau'_{\{v,w\}}}\right),$$

[4] $\tau_v, \tau_{\{v,w\}}$ are **physical continuous-time** delays.

where d_v is the degree of node v and $\tau'_{\{v,w\}} = \tau_{\{v,w\}} + \max(\tau_v, \tau_w)$. Let Λ be the spectral gap (smallest non-null eigenvalue of the weighted Laplacian) of the graph G with weights

$$\lambda_{\{v,w\}} = \frac{\min_{u \sim \{v,w\}} p_{\{v,w\}}}{d \sum_{e \in \mathcal{E}} p_e}, \ \{v,w\} \in \mathcal{E}\,,$$

where d is the max degree in the graph. Then, Assumption 5.3.2 is verified for $\frac{\rho}{k_\rho} = \tilde{\mathcal{O}}(\Lambda)$.

Given a graph G with physical communication and computation latencies $\{\tau_v, \tau_{\{v,w\}}\}$ (Assumption 5.5.1), we are thus able to exhibit a communication scheme that satisfies communication and computation constraints, while still fitting in the framework of AGRAF SGD under the assumptions used in our convergence rates. Crucially, the mixing constant Λ explicitly depends on the graph and the delays, through the smallest non-null eigenvalue of the weighted graph Laplacian, with explicit weights $\lambda_{\{v,w\}}$ on the edges. These weights depend on **local** delays: having straggler nodes or edges do not slow down communication or computations, if there are fast edges/nodes that are dense enough in the graph. To further highlight the importance of having weights $\lambda_{\{v,w\}}$ that only depend on the local delays, this can be put in perspective of Asynchronous SGD, that is proved to depend only on the averaged computation delay $\frac{1}{n}\sum_{v \in \mathcal{V}} \frac{1}{\tau_v}$ rather than the max delay [KSJ22, MIMA18]. For decentralized optimization over a given graph, depending on the averaged communication delays wouldn't make sense since all communication paths need to be taken into account; hence, the counterpart to the mean delay in the graph is a **weighted** Laplacian, with weights on edge $\{v,w\}$ that are function of **local** delays, instead of a max delay which is the asynchronous speedup [EHM21a].

Our proof of Theorem 5.5 follows three main steps: *i)* Deriving convergence results for more general communication schemes than loss networks, under deterministic assumptions on the activations. *ii)* Adapting Step i) to stochastic assumptions on the delays. *iii)* Deriving high-probability upper-bounds on the delays between two activations in loss networks in order to fall under the assumptions of Step i).

Conclusion

We introduced a unifying framework for studying asynchronous and decentralized algorithms; our analysis recovers and improves over that of previous asynchronous decentralized SGD works, while being much more general. The flexibility of our framework furthermore enables us to leverage an asynchronous speedup under communication and computation delays, by the introduction of Loss Networks and new analysis tools, thus providing a non-trivial sampling scheme that still satisfies the ergodic mixing property introduced by [KLB+20].

Appendix of Chapter 5

5.A. Equivalence of two ergodic mixing assumptions

The following assumption is a consequence of Assumption 5.3.2: if Assumption 5.3.2 holds for some τ, ρ, then Assumption 5.A.1 holds for $\bar{\rho} = c\frac{\rho}{\tau}$ where c is some numerical constant. In fact, as we prove in Proposition 5.A.1, they are both equivalent, but the following proves to be easier to handle in the analysis.

Assumption 5.A.1. *$W_k\mathbb{1} = \mathbb{1}$ and there exist $\bar{\rho}$ such that we have $\forall k, \ell \in \mathbb{N}$ and $\forall \mathbf{x} \in \mathbb{R}^{\mathcal{V}}$:*

$$\begin{aligned}\mathbb{E}\left[\left\|W^{(k:k+\ell)}\mathbf{x} - \frac{1}{n}\mathbb{1}\mathbb{1}^\top\mathbf{x}\right\|^2 |\mathcal{F}_k\right] \\ \leq 2(1-\bar{\rho})^{2\ell}\left\|\mathbf{x} - \frac{1}{n}\mathbb{1}\mathbb{1}^\top\mathbf{x}\right\|^2 .\end{aligned} \tag{5.9}$$

Proposition 5.A.1. *Assumptions 5.3.2 and 5.A.1 are equivalent, in the following sense.*

1. *If Assumption 5.3.2 holds for some $\rho \in [0,1]$ and for some $k_\rho \in \mathbb{N}^*$, then Assumptions 5.A.1 holds for $\bar{\rho} = c\frac{\rho}{k_\rho}$, for $c > 0$ some numerical constant.*
2. *If Assumption 5.A.1 holds for some $\bar{\rho} \in [0,1]$, then Assumption 5.3.2 holds for any $\rho \in (0,1)$ and $k_\rho = \left\lceil \frac{\frac{1}{2}\ln(2)\ln(1-\rho)}{\ln(1-\bar{\rho})} \right\rceil$ ($\propto \frac{\rho}{\bar{\rho}}$ for $\rho, \bar{\rho}$ small).*

Proof. We first prove **1.** Assume that Assumption 5.3.2 holds for some ρ, k_ρ. If Assumption 5.3.2 holds for ρ it holds for any $\rho' < \rho$, so that we can assume without loss of generality that $\rho \leq 1 - \sqrt{2}$. Let $k, \ell \in \mathbb{N}$ and $\mathbf{x} \in \mathbb{R}^{\mathcal{V}}$. Using Assumption 5.3.2 $\lfloor \frac{\ell}{k_\rho} \rfloor$, we have that:

$$\mathbb{E}\left[\left\|W^{(k:k+\ell)}\mathbf{x} - \frac{1}{n}\mathbb{1}\mathbb{1}^\top\mathbf{x}\right\|^2 |W_0, \dots, W_k\right] \leq (1-\rho)^{2\lfloor \ell/k_\rho \rfloor}\left\|\mathbf{x} - \frac{1}{n}\mathbb{1}\mathbb{1}^\top\mathbf{x}\right\|^2 .$$

Thus, $(1-\rho)^{2\lfloor \ell/k_\rho \rfloor} \leq (1-\rho)^{2(\ell/k_\rho - 1)} \leq \frac{1}{(1-\rho)^2}(1-\rho)^{2\ell/k_\rho}$. Then, $\frac{1}{(1-\rho)^2} \leq 2$ and $(1-\rho)^{2\ell/k_\rho} \leq e^{-2\ell\rho/k_\rho} \leq (1 - c\frac{\rho}{k_\rho})^{2\ell}$ for $c \in (0,1)$ some numerical constant ($c = \frac{e-1}{e}$), since $\frac{\rho}{k_\rho} \leq 1$.

We now prove **2.** Assume that Assumption 5.A.1 holds for $\bar{\rho} > 0$, and let $\rho > 0$. We have:

$$\mathbb{E}\left[\left\|W^{(k:k+\ell)}\mathbf{x} - \frac{1}{n}\mathbb{1}\mathbb{1}^\top\mathbf{x}\right\|^2 |W_0, \dots, W_k\right] \leq (1-\rho)^2\left\|\mathbf{x} - \frac{1}{n}\mathbb{1}\mathbb{1}^\top\mathbf{x}\right\|^2 ,$$

provided that ℓ satisfies:

$$2(1-\bar{\rho})^{2\ell} \leq (1-\rho)^2 .$$

This is satisfied for:

$$\ell \geq \frac{\frac{1}{2}\ln(2)\ln(1-\rho)}{\ln(1-\bar{\rho})} ,$$

and thus Assumption 5.5 holds for ρ and $k_\rho = \left\lceil \frac{\frac{1}{2}\ln(2)\ln(1-\rho)}{\ln(1-\bar{\rho})} \right\rceil$. □

5.B. Preliminaries for our convergence rates

For $k \geq 0$, , and for any $k \geq 0$ and $v \in \mathcal{V}$:

$$\begin{aligned}&\mathrm{next}(k,v) = \inf\left\{\ell \geq k\,,\ v \in \mathcal{I}_\ell\right\}, \quad \mathrm{prev}(k,v) = \sup\left\{\ell < k\,,\ v \in \mathcal{I}_\ell\right\} \cup \{0\}\,,\\ &\tau(k,v) = k - \mathrm{prev}(k+1,v)\,.\end{aligned}$$

In other words, at a given iteration k, $\mathrm{next}(k,v)$ is the iteration at which the node v will finish computing its current gradient, $\mathrm{prev}(k,v)$ is the iteration at which the node v started computing its current gradient, and $\tau(k,v)$ is the current computational delay of node v at time k.

Let also $\bar{x}^k = \frac{1}{n}\sum_{v\in\mathcal{V}} x_v^k \in \mathbb{R}^d$ and $\mathbf{g}^k = (\mathbb{1}_{v\in\mathcal{I}_k}\nabla F_v(x_v^{\mathrm{prev}(k,v)}, \xi_v^{\mathrm{prev}(k,v)}))$, so that $\mathbf{x}^{k+1} = W_k\mathbf{x}^k - \gamma\mathbf{g}^k$.

5.B.1. Virtual iterate sequence to handle delays

As in [MBEW22], the delay analysis relies on the study of a virtual sequence. Noticing that $\bar{x}^{k+1} = \bar{x}^k - \frac{\gamma}{n}\sum_{v\in\mathcal{I}_k} g_v^{t-\tau(k,v)}$ and mimicking the analysis of asynchronous SGD, we introduce the sequence $\left\{\hat{x}^k, k \geq 1\right\}$ that lives in $\mathbb{R}^d$, defined through the following recursion:

$$\hat{x}^{k+1} = \hat{x}^k - \frac{\gamma}{n}\sum_{v\in\mathcal{I}_k} g_v^k\,, \quad \hat{x}_1 = \bar{x}_0 - \frac{\gamma}{n}\sum_{v\in\mathcal{V}} g_v^0\,.$$

We then have, for all $k \geq 1$:

$$\hat{x}^k - \bar{x}^k = -\frac{\gamma}{n}\sum_{v\in\mathcal{V}\setminus\mathcal{I}_k} g^{\mathrm{prev}(k,v)}\,.$$

The difference $\left\|\hat{x}^k - \bar{x}^k\right\|$ can thus be easily bounded.

Lemma 5.B.1 (Virtual iterates control)**.** *If stochastic gradients are bounded by a constant $B > 0$, we have:*

$$\left\|\hat{x}^k - \bar{x}^k\right\| \leq \gamma B\,. \tag{5.10}$$

In the general case,

$$\mathbb{E}\left[\left\|\hat{x}^k - \bar{x}^k\right\|^2\right] \leq \frac{2\gamma^2}{n}\left(\sigma^2 + \sum_{v\in\mathcal{V}}\mathbb{E}\left[\left\|\nabla f_v(x_v^{\mathrm{prev}(v,k)})\right\|^2\right]\right). \tag{5.11}$$

Proof. Equation (5.10) is proved using a triangle inequality, while Equation (5.11) is a direct application of [SK20, Lemma 15]. □

5.B.2. Consensus control

Lemma 5.B.2 (Consensus control)**.** *We have:*

$$\sum_{k<K}\mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 \leq 2\gamma^2\sigma^2\bar{\rho}^{-1}\sum_{k<K}|\mathcal{I}_k| + \frac{4\gamma^2}{\bar{\rho}^2}\sum_{k<K}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[\left\|\nabla f_v(x_v^{k-\tau(k,v)})\right\|^2\right] \tag{5.12}$$

$$\leq 2\gamma^2\sigma^2\bar{\rho}^{-1}\sum_{k<K}|\mathcal{I}_k| + \frac{4\gamma^2}{\bar{\rho}^2}\sum_{k<K}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\right]. \tag{5.13}$$

If the stochastic gradients are bounded by some $B > 0$,

$$\sum_{k<K} \mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 \leq \frac{2\gamma^2 B^2}{\bar{\rho}^2} \sum_{k<K} |\mathcal{I}_k| \,. \tag{5.14}$$

Proof. Under Assumption 5.3.2, we can bound the variations of $\mathbf{x}^k - \bar{\mathbf{x}}^k$ (here, $\bar{\mathbf{x}}^k = \mathbb{1}\mathbb{1}^\top \mathbf{x}^k$). Using Cauchy-Schwarz inequality, for $a_m > 0$ scalars and $b_m \in \mathbb{R}^p$ vectors, we have:

$$\left\|\sum_m b_m\right\|^2 \leq \left(\sum_m a_m^{-1}\right)\left(\sum_m a_m \|b_m\|^2\right) .$$

We now apply this to $\mathbf{x}^k - \bar{\mathbf{x}}^k = -\gamma \sum_{m=0}^k W^{(m:k)}(\tilde{\mathbf{g}}^m - \bar{\tilde{\mathbf{g}}}^m)$ to obtain:

$$\begin{aligned}
\mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 &= \mathbb{E}\left[\left\|\gamma \sum_{m=0}^k W^{(m:k)}(\tilde{\mathbf{g}}^m - \bar{\tilde{\mathbf{g}}}^m)\right\|^2\right] \\
&\leq \gamma^2 \sum_{m'=0}^k (1-\bar{\rho})^{k-m'} \sum_{m=0}^k (1-\bar{\rho})^{-(k-m)} \mathbb{E}\left[\left\|W^{(m:k)}(\tilde{\mathbf{g}}^m - \bar{\tilde{\mathbf{g}}}^m)\right\|^2\right] \\
&\leq 2\gamma^2 \frac{1}{\bar{\rho}} \sum_{m=0}^k (1-\bar{\rho})^{k-m} \mathbb{E}\left[\|\tilde{\mathbf{g}}^m\|^2\right]
\end{aligned}$$

leading to, if stochastic gradients are bounded by B:

$$\mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 \leq \frac{2\gamma^2 B^2}{\bar{\rho}} \sum_{\ell<k} (1-\bar{\rho})^{k-\ell} |\mathcal{I}_\ell| \,,$$

and thus:

$$\sum_{k<K} \mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 \leq \frac{2\gamma^2 B^2}{\bar{\rho}^2} \sum_{k<K} |\mathcal{I}_k| \,.$$

We also have, using a bias-variance decomposition (not exactly, since the $\mathbf{g}^m$ are not independent, but using the martingale version as in [SK20, Lemma 15]):

$$\begin{aligned}
\mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 &= \mathbb{E}\left[\left\|\gamma \sum_{m=0}^k W^{(m:k)}(\tilde{\mathbf{g}}^{m-\tau(m)} - \bar{\tilde{\mathbf{g}}}^m)\right\|^2\right] \\
&\leq 2\gamma^2\sigma^2 \sum_{\ell<k} (1-\bar{\rho})^{k-\ell} |\mathcal{I}_\ell| + \frac{4\gamma^2}{\bar{\rho}} \sum_{m=0}^k (1-\bar{\rho})^{k-m} \sum_{v\in\mathcal{I}_m} \mathbb{E}\left[\left\|\nabla f_v(x_v^{(m-\tau(m,v))})\right\|^2\right] ,
\end{aligned}$$

so that:

$$\sum_{k<K} \mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 \leq 2\gamma^2\sigma^2\bar{\rho}^{-1} \sum_{k<K} |\mathcal{I}_k| + \frac{4\gamma^2}{\bar{\rho}^2} \sum_{k<K} \sum_{v\in\mathcal{I}_k} \mathbb{E}\left[\left\|\nabla f_v(x_v^{k-\tau(k,v)})\right\|^2\right] .$$

□

5.C. Loss Networks analysis

In this section, we prove Theorem 5.5 and provide some more information on loss networks. The updates of *decentralized SGD on loss networks* write as:

$$\begin{cases} x_{v_k}^{k+1} = \dfrac{x_{v_k}^k + x_{w_k}^k}{2} - \gamma \nabla F_{v_k}\left(x_{v_k}^{k-\tau(v_k,k)}, \xi_{v_k}^{k-\tau(v_k,k)}\right) \\ x_{w_k}^{k+1} = \dfrac{x_{v_k}^k + x_{w_k}^k}{2} - \gamma \nabla F_{w_k}\left(x_{w_k}^{k-\tau(w_k,k)}, \xi_{w_k}^{k-\tau(w_k,k)}\right) \end{cases}, \tag{5.15}$$

leading to $\mathbf{x}^{k+1} = W_k \mathbf{x}^k - \gamma \mathbf{g}^k$, for $W_k = W_{\{v_k,w_k\}} = I_{\mathcal{V}} - \frac{(e_{v_k} - e_{w-k})(e_{v_k} - e_{w-k})^\top}{2}$, and $\mathbf{g}^k$ the corresponding delayed gradients. Note then that this takes the same form as the **AGRAF SGD** sequence.

Definition 5.C.1 (Poisson point process (P.p.p.))**.** *A* **Poisson point process** *of intensity $p > 0$ is a random discrete subset $\mathcal{P}$ of $\mathbb{R}_{\geq 0}$ that can be written as $\mathcal{P} = \{T_0 < T_1 < \ldots < T_k < \ldots\}$, where $(T_k - T_{k-1})_{k\geq 1}$ are i.i.d. exponential random variables of mean $\frac{1}{p}$.*

[BGPS06] consider a model (without any delay) for gossip algorithms, where updates are that of Equation (5.15) without the gradient steps, and these updates happen at the times of Poisson point processes (a *P.p.p.* of intensity $p_{\{v,w\}}$ for an update along $\{v,w\}$). Consequently, W_k is independent from the past, and $\mathbb{P}\left(W_k = W_{\{v,w\}}\right) \propto p_{\{v,w\}}$.

The *P.p.p. model* considered in [BGPS06] where the updates are performed at the times of *Poisson point processes* is particularly amenable to analysis, but it assumes that communications and computations are done instantaneously. Thus, actual implementations differ from its underlying assumptions, unless further synchrony is assumed. To alleviate this issue, with pairwise communications ruled by point processes as a baseline, we consider a protocol in which nodes are tagged as *busy* when they are already engaged in an update, and communications between busy nodes are forbidden. Our model is inspired from classical Loss Network models [Kel91b], in which edges are activated following the same procedure as in the *P.p.p. model*, with a *P.p.p.* of intensity $p_{\{v,w\}}$. Note that we do not consider these intensities to be constraints of the problem, but rather parameters of the algorithm, that can be tuned. Each node has an exponential clock of intensity $p_v \frac{1}{2} = \sum_{w\sim v} p_{\{v,w\}}$. At each clock-ticking, if v is not busy, it selects a neighbor w with probability $p_{\{v,w\}} / \sum_{u\sim v} p_{\{u,v\}}$. If w is not *busy*, v and w compute and exchange information, becoming busy for a duration $\tau'_{\{v,w\}}$. We can think of this procedure as classical gossip on an underlying random graph that follows a Markov-Chain process. The difference between our communication model on Loss Networks and the P.p.p. model lies in that in our case, W_k is not independent on the past. In fact, we have:

$$\mathbb{P}\left(\{v_k, w_k\} = \{v,w\} | \mathcal{F}_k\right) = \frac{\mathbb{1}_{\{v,w \text{ not busy at time } T_k\}} p_{\{v,w\}}}{\sum_{\{u,u'\}\in\mathcal{E}} \mathbb{1}_{\{u,u' \text{ not busy at time } T_k\}} p_{\{u,u'\}}},$$

leading to complicated intricacies between the matrices $(W_k)_k$, that we need to handle.

Proving Theorem 5.5 requires to show that there exist ρ, k_ρ (that need to be computed) such that for any $k \geq 0$, $\mathbf{x} \in \mathbb{R}^{\mathcal{V}}$,

$$\mathbb{E}\left[\left\| W_{\{v_{k+k_\rho-1}, w_{k+k_\rho-1}\}} \cdot \ldots \cdot W_{\{v_k,w_k\}} (\mathbf{x} - \bar{\mathbf{x}}) \right\|^2 | \mathcal{F}_k \right] \leq (1-\rho)^2 \|\mathbf{x} - \bar{\mathbf{x}}\|^2 .$$

Our proof of Theorem 5.5 follows three main steps: *i)* Deriving convergence results for more general communication schemes than loss networks, under deterministic assumptions on the activations. *ii)* Adapting Step i) to stochastic assumptions on the delays. *iii)* Deriving high-probability upper-bounds on the delays between two activations in loss networks in order to fall under the assumptions of Step i).

5.C.1. Descent lemma under deterministic assumptions on the activations

We consider general activation processes $\mathcal{P}_{\{v,w\}}$, where we define $\mathcal{P}_{\{v,w\}}$ as $\mathcal{P}_{\{v,w\}} = \{T_k : \{v_k, w_k\} = \{v, w\}\}$, and these times are called **activation times of edge {v,w}**. When edge $\{v, w\}$ is activated, the update described in (5.15) is performed. The delay of an edge is defined as its (random) waiting time between two activations. Two ergodicity-like conditions on the delays are needed: *(i) edges activated regularly enough and (ii) incident edges must not be activated too many times.*

We now formally introduce these assumptions. We consider discrete time in this section: more precisely, $k \in \mathbb{N}$ stands for the k-th edge activation.

Definition 5.C.2. *Consider a communication scheme with edge-activation point processes $\mathcal{P}_{\{v,w\}}$. Let $k = 0, 1, 2, ...$ index the consecutive edge activations. Let $\ell \in \mathbb{N}$, $\{v, w\}$ and $\{u, u'\} \in E$. Let $k_{\{v,w\}} < \ell_{\{v,w\}}$ such that $k_{\{v,w\}} \leq k < \ell_{\{v,w\}}$ be consecutive activation times (in discrete time) of $\{v, w\}$. Denote $T_{\{v,w\}}(k) = \ell_{\{v,w\}} - k_{\{v,w\}} - 1$ the total number of edge activations between the two consecutive activations of $\{v, w\}$. Denote $N(\{u, u'\}, \{v, w\}, k)$ the number of activations of edge $\{v, w\}$ in the activations $\{s_{\{v,w\}}, s_{\{v,w\}} + 1, ..., t_{\{v,w\}} - 1\}$.*

Assumption 5.C.1 (Delay Assumptions)**.** *There exist $T \in \mathbb{N}^*$, $a, b > 0$, and $\ell_{\{v,w\}} > 0, \{v, w\} \in E$ such that, for the quantities and the communication scheme in Definition 5.C.2:*

1. *For all $k \in \mathbb{N}$, all edges are activated between iterations k and $k + T - 1$.*
2. *$\forall k \geq 0, \forall(\{v, w\}) \in E, T_{\{v,w\}}(k) \leq a\ell_{\{v,w\}}$: $(\{v, w\})$ is activated at least every $a\ell_{\{v,w\}}$ activations.*
3. *$\forall k \geq 0, \forall(\{v, w\}), (\{u, u'\}) \in E$ such that $(\{u, u'\}) \sim (\{v, w\})$, $N(\{u, u'\}, \{v, w\}, k) \leq \lceil \frac{b\ell_{\{v,w\}}}{\ell_{\{u,u'\}}} \rceil$.*

Assumption (1) is implied by Assumption (2) if $T = \max_{(\{v,w\})} \ell_{\{v,w\}}$. Taking $\ell_{\{v,w\}}$ as a deterministic upper-bound on the delays of edge $(\{v, w\})$ between two activations in continuous time is sufficient to have Assumption (2) and (3), with some normalizing constant a, and b such that $\ell_{\{v,w\}}/b$ is a lower-bound on these delays.

The main technical difficulty lies in the fact that at a defined activation time t, some nodes are not available: at any time $k \geq 0$, $\sum_{(\{v,w\})\in E \text{ not busy}} W_k$ usually differs from $\sum_{\{v,w\}} p_{\{v,w\}} W_{\{v,w\}}$ (and $\sum_{(\{v,w\})\in E \text{ not busy}} W_k$ may have a null spectral gap) as in *Markov-Chain Gradient Descent* [Eve23], thus making an analysis such as in the *P.p.p. model* impossible. To alleviate this difficulty, in order to make sure that all edges are taken into account when performing the averaging, the Lyapunov function Λ_k that we study considers the value of the objective for T consecutive activation times. It is defined as follows:

$$\forall k \in \mathbb{N}, \Lambda_k(\mathbf{x}) = \frac{1}{T} \sum_{\ell=k}^{k+T-1} \left\| W^{(0,\ell)}(\mathbf{x} - \bar{\mathbf{x}}) \right\|^2, \quad \mathbf{x} \in \mathbb{R}^{\mathcal{V}}.$$

The first step of the proof of Theorem 5.5 consists in proving the following.

Theorem 5.6. *Consider a general communication scheme as in Definition 5.C.2, that satisfies Assumption 5.C.1 for constants $\ell_{\{v,w\}}, a, b > 0$,. Let γ be the smallest positive eigenvalue of the Laplacian of the graph G with weights:*

$$\nu_{\{v,w\}} = C\ell^{-1}_{\{v,w\}} \min_{\{u,u'\}\sim\{v,w\}} \frac{\ell_{\{u,u'\}}}{\ell_{\{v,w\}}}, \quad \{v, w\} \in \mathcal{E},$$

where $C = \frac{1}{2a+8d^2_{\max}ab}$. Then we have, for all $k, \ell \in \mathbb{N}$:

$$\Lambda_{k+\ell}(\mathbf{x}) \leq (1 - \gamma)^{\ell} \Lambda_k(\mathbf{x}).$$

Proof. We fix $\mathbf{x} \in \mathbb{R}^{\mathcal{V}}, k, \ell$. To prove this intermediate theorem, we need to study every matrix multiplication involved. At iteration k, not every coordinates is available, hence the need to study the impact of T multiplications together.

A gradient step alongside edge $\{v, w\}$ only involves edges in its neighborhood (thanks to the sparsity of the matrix A), a key element that will need to be explicited. The proof involves three main steps.

Before that, we need to introduce **edge dual variables**. Matrix multiplications by matrices like $W_{\{v,w\}}$ aim at minimizing the function $F(\mathbf{y}) = \frac{1}{2}\sum_{v\in\mathcal{V}}(y_v - x_v)^2$, which is minimized at $\mathbf{y} = \bar{\mathbf{x}}$. A standard way to deal with the constraint $x_1 = ... = x_n$, is to use a dual formulation, by introducing a dual variable $\lambda \in \mathbb{R}^{\mathcal{E}}$ indexed by the edges. We first introduce a matrix $A \in \mathbb{R}^{\mathcal{V}\times\mathcal{E}}$ such that $\mathrm{Ker}(\mathrm{A}^\top) = \mathrm{Vect}(\mathbb{I})$ where $\mathbb{I}$ is the constant vector $(1, ..., 1)^\top$. A is chosen such that:

$$\forall \{v, w\} \in E, Ae_{\{v,w\}} = \mu_{\{v,w\}}(e_v - e_w). \tag{5.16}$$

for some non-null constants $\mu_{\{v,w\}}$. We define $\mu_{\{v,w\}} = -\mu_{\{v,w\}}$ for this writing to be consistent. This matrix A is a square root of the laplacian of the graph weighted by $\nu_{\{v,w\}} = \mu^2_{\{v,w\}}$. The constraint $x_1 = ... = x_n$ can then be written $A^\top x = 0$. The dual problem reads as follows:

$$\min_{\mathbf{y}\in\mathbb{R}^{\mathcal{V}}, A^\top\mathbf{y}=0} F(\mathbf{y}) = \min_{\mathbf{y}\in\mathbb{R}^{\mathcal{V}}} \max_{\lambda\in\mathbb{R}^{\mathcal{E}}} F(\mathbf{y}) - \langle A^\top\mathbf{y}, \lambda\rangle.$$

Let $F^*_A(\lambda) := F^*(A\lambda) = F_A(\lambda)$ for $\lambda \in \mathbb{R}^{E\times d}$ where F^* is the Fenchel conjugate of F. Now, notice that for our particular form of F, we in fact have $F^* = F$. The dual problem reads

$$\min_{\mathbf{y}\in\mathbb{R}^{\mathcal{V}}, y_1=...=y_n} F(\mathbf{y}) = \max_{\lambda\in\mathbb{R}^{\mathcal{E}}} -F_A(\lambda).$$

Thus $F^*_A(\lambda)$ is to be minimized over the dual variable $\lambda \in \mathbb{R}^{\mathcal{E}}$.

We now make a parallel between pairwise operations between adjacent nodes in the network and coordinate gradient steps on F^*_A. As $F^*_A(\lambda) = \max_{\mathbf{y}\in\mathbb{R}^{\mathcal{V}}} -F(\mathbf{y}) + \langle A\lambda, \mathbf{y}\rangle$, to any $\lambda \in \mathbb{R}^{\mathcal{E}}$ a primal variable $\mathbf{y} \in \mathbb{R}^{\mathcal{V}}$ is uniquely associated through the formula $\nabla F(\mathbf{y}) = A\lambda$. The partial derivative of F^*_A with respect to coordinate $\{v, w\} \in \mathcal{E}$ of λ reads :

$$\nabla_{\{v,w\}} F^*_A(\lambda) = (Ae_{\{v,w\}})^\top \nabla F^*(A\lambda) = \mu_{\{v,w\}}(\nabla g^*_v((A\lambda)_v) - \nabla g^*_w((A\lambda)_w)),$$

where we denote $g_v(y) : \frac{1}{2}(y - x_v)^2$. Consider then the following step of coordinate gradient descent for F^*_A on coordinate $\{v, w\}$ of λ, performed when edge $\{v, w\}$ is activated at iteration k (corresponding to time T_k), and where $U_{\{v,w\}} = e_{\{v,w\}}e^\top_{\{v,w\}}$:

$$\lambda_{k+1} = \lambda_{k+1} - \frac{1}{\mu^2_{\{v,w\}}} U_{\{v,w\}} \nabla_{\{v,w\}} F^*_A(\lambda_k). \tag{5.17}$$

Denoting $\mathbf{y}_k = A\lambda_k \in \mathbb{R}^{\mathcal{V}}$, we obtain the following formula for updating coordinates v and w of $\mathbf{y}$ when $\{v, w\}$ activated:

$$y_{v,k+1} = y_{v,k} - \frac{1}{2}(y_{v_k} - y_{w_k}) = \frac{1}{2}(y_{v_k} y_{w_k}) = y_{w,k+1}\,. \tag{5.18}$$

Thus, $\mathbf{y}^{k+1} = W_k\mathbf{y}^k$ is equivalent to $\lambda_{k+1} = \lambda_k - \frac{1}{2\mu^2_{\{v_k,w_k\}}}\nabla_{\{v_k,w_k\}} F^*_A(\lambda_k)$, which is easier to study. Also, notice that this is the consensus distance exctly: $F^*_A(\lambda) = F(\mathbf{y})$ for $\mathbf{y} = A\lambda$.

Hence, $\Lambda_k(\mathbf{x}) = F(\mathbf{y}^k) = F^*_A(\lambda_k)$ here $\mathbf{y}^k = A\lambda^k$ is obtained with the recursion $\lambda^{k+1} = \lambda^k - \frac{1}{2\mu^2_{\{v_k,w_k\}}}\nabla_{\{v_k,w_k\}} F^*_A(\lambda^k)$, with initialisation $\mathbf{y}^0 = \mathbf{x}$: we thus study this sequence.

Step 1: First, notice that F_A^* is $\mu^2_{\{v,w\}}$-smooth along every coordinate $\{v,w\}$, so that using local smoothness, for all $\{v,w\} \in \mathcal{E}$ and $\lambda \in \mathbb{R}^{\mathcal{E}}$, for $\gamma \leq \frac{1}{2\mu^2_{\{v_k,w_k\}}}$, we have:

$$F_A^*(\lambda - \nabla_{\{v,w\}} F_A^*(\lambda)) - F_A^*(\lambda) \leq \frac{1}{4\mu^2_{\{v,w\}}} \left\| \nabla_{\{v,w\}} F_A^*(\lambda) \right\|^2 . \tag{5.19}$$

Applying Equation (5.19), where $\{v_\ell, w_\ell\}$ is the ℓ^{th} activated edge:

$$F_A^*(\lambda^{\ell+1}) - F_A^*(\lambda^\ell) \leq -\frac{1}{4\mu^2_{\{v_\ell,w_\ell\}}} \|\nabla_{\{v_\ell,w_\ell\}} F_A^*(\lambda^\ell)\|^2 . \tag{5.20}$$

Hence, summing:

$$\Lambda_{k+1} \leq \Lambda_k - \frac{1}{T} \sum_{k \leq \ell < k+T} \frac{1}{4\mu^2_{\{v_\ell,w_\ell\}}} \|\nabla_{\{v_\ell,w_\ell\}} F_A^*(\lambda^\ell)\|^2 , \tag{5.21}$$

Notice that:

$$\frac{1}{T} \sum_{k \leq \ell < k+T} \sum_{\{v,w\} \in \mathcal{E}} \|\nabla_{\{v,w\}} F_A^*(\lambda^\ell)\|^2 = \frac{1}{T} \sum_{k \leq \ell < k+T} \|\nabla F_A^*(\lambda^\ell)\|^2 \geq \sigma_A \Lambda_t \tag{5.22}$$

σ_A is the strong convexity parameter of F_A^* which is equal tolower bounded by $\lambda^+_{min}(A^T A)$, which itself is exactly the smallest positive non-null eigenvalue of the graph Laplacian with weights $\mu^2_{\{v,w\}}$. Hence, if an inequality of the type

$$\frac{C}{T}\frac{1}{T} \sum_{k \leq \ell < k+T} \sum_{\{v,w\} \in \mathcal{E}} \|\nabla_{\{v,w\}} F_A^*(\lambda^\ell)\|^2 \leq \frac{1}{4\mu^2_{\{v_\ell,w_\ell\}}} \|\nabla_{\{v_\ell,w_\ell\}} F_A^*(\lambda^\ell)\|^2 \tag{5.23}$$

holds, we have using strong convexity:

$$\Lambda_{k+1} \leq \Lambda_k - \frac{C}{T} \sum_{k \leq \ell < k+T} \|\nabla F_A^*(\lambda^\ell)\|^2 \leq (1 - C\sigma_A)\Lambda_k . \tag{5.24}$$

We thus need to tune correctly the $\mu^2_{\{v,w\}}$ and C in order to have (5.23) verified.

Step 2: We are looking for necessary conditions for (5.23) to hold. In the left term, every coordinate is present at each time ℓ. However, in the right hand side of the inequality, just the activated one is present. We will need to compensate this with a bigger factor in front of the gradients. In order to compare these quantities, we need to introduce upper bound inequalities on $\|\nabla_{\{v,w\}} F_A^*(\lambda(s))\|^2$, that only make activated coordinates intervene. Let $s \in \{t, ..., t+T-1\}$, and suppose that there exists $t \leq r \leq s < r + t_{\{v,w\}} \leq t+T-1$ such that $\{v,w\}$ is activated at times r and $r + t_{\{v,w\}}$. Thanks to the asumption on T, either one of these integers exists. If the other one doesn't, replace it with t for r, and by $t+T-1$ for $r + t_{\{v,w\}}$. Thanks to our asumptions, we know that $t_{\{v,w\}} \leq a\ell_{\{v,w\}}$. We have the following basic inequalities:

$$\begin{aligned} \|\nabla_{\{v,w\}} F_A^*(\lambda(s))\|^2 &\leq (\|\nabla_{\{v,w\}} F_A^*(\lambda(r))\| + \|\nabla_{\{v,w\}} F_A^*(\lambda(s)) - \nabla_{\{v,w\}} F_A^*(\lambda(r))\|)^2 \\ &\leq 2(\|\nabla_{\{v,w\}} F_A^*(\lambda(r))\|^2 + \|\nabla_{\{v,w\}} F_A^*(\lambda(s)) - \nabla_{\{v,w\}} F_A^*(\lambda(r))\|^2). \end{aligned}$$

The quantity $\|\nabla_{\{v,w\}} F_A^*(\lambda(s)) - \nabla_{\{v,w\}} F_A^*(\lambda(r))\|^2$ then needs to be controlled. We use the following lemma.

Lemma 5.C.1. *For $\lambda, \lambda' \in R^{\mathcal{E}}$, and $\{v,w\} \in E$, we have:*

$$\|\nabla_{\{v,w\}} F_A^*(\lambda) - \nabla_{\{v,w\}} F_A^*(\lambda')\|^2 \tag{5.25}$$

$$\leq 8 d_{\{v,w\}} \mu_{\{v,w\}}^2 \sum_{(\{u,u'\})\sim(\{v,w\})} \mu_{\{u,u'\}}^2 \|\lambda_{\{u,u'\}} - \lambda'_{\{u,u'\}}\|^2. \tag{5.26}$$

Proof. First, notice that $\nabla_{\{v,w\}} F_A^*(\lambda) = \mu_{\{v,w\}}(\nabla g_i^*((A\lambda)_v) - \nabla g_j^*((A\lambda)_w))$. Then:

$$\begin{aligned}
\|\nabla f_v^*((A\lambda)_v) - \nabla f_v^*((A\lambda')_w)\| &= \|(A(\lambda - \lambda'))_v\| \text{ (smoothness)} \\
&= \| \sum_{\{u,u'\}\sim\{v,w\}} \mu_{\{u,u'\}} (\lambda - \lambda')_{\{u,u'\}} \| \\
&\leq \sum_{\{u,u'\}\sim\{v,w\}} \mu_{\{u,u'\}} \|(x - x')_{\{u,u'\}}\|
\end{aligned}$$

Conclude by taking the square and summing for v and w. □

Using this with $\lambda = \lambda(s)$ and $\lambda' = \lambda(r)$:

$$\|\nabla_{\{v,w\}} F_A^*(\lambda(s))\|^2 \leq 2\|\nabla_{\{v,w\}} F_A^*(\lambda(r))\|^2 \tag{5.27}$$

$$+ 2 d_{\{v,w\}} \sum_{r<k<r+t_{\{v,w\}}} N((\{v_k, w_k\}), \{v,w\}, k) \frac{\mu_{\{v,w\}}^2}{2\mu_{\{v_k,w_k\}}^2} \|\nabla_{\{v_k,w_k\}} F_A^*(\lambda(k))\|^2 \tag{5.28}$$

$$\leq 2\|\nabla_{\{v,w\}} F_A^*(\lambda(r))\|^2 \tag{5.29}$$

$$+ 2 d_{\{v,w\}} \sum_{r<k<r+t_{\{v,w\}}} \left\lceil b \frac{\ell_{\{v,w\}}}{L_{\{v_k,w_k\}}} \right\rceil \frac{\mu_{\{v,w\}}^2}{\mu_{\{v_k,w_k\}}^2} \|\nabla_{\{v_k,w_k\}} F_A^*(\lambda(k))\|^2 \tag{5.30}$$

The advantage of this last expression is that only activated quantities are present on the right hand side.

Step 3: The last step of the proof consists in summing the last inequality for $t \leq \ell < t+T$, $\{v,w\} \in E$. When summing, each $\|\nabla_{\{v_k,w_k\}} F_A^*(\lambda(k))\|^2$ appears on the right hand-side of the inequality, with a factor upper-bounded by (here instead of $\{v_k, w_k\}$ we write $(\{v,w\})$):

$$2a\ell_{\{v,w\}} + 2 d_{\{v,w\}} \sum_{\{u,u'\}\sim\{v,w\}} a\ell_{\{u,u'\}} \left\lceil \frac{b\ell_{\{u,u'\}}}{\ell_{\{v,w\}}} \right\rceil \frac{\mu_{\{u,u'\}}^2}{\mu_{\{v,w\}}^2}. \tag{5.31}$$

We want the expression above multiplied by C defined in Step 1 to be upper-bounded by $\frac{1}{4\mu_{\{v,w\}}^2}$, in order for (5.23) to be verified. This is possible if and only if:

$$C\left(4a\ell_{\{v,w\}}\mu_{\{v,w\}}^2 + 4 d_{\{v,w\}} \sum_{\{u,u'\}\sim\{v,w\}} a \left\lceil \frac{b\ell_{\{u,u'\}}}{\ell_{\{v,w\}}} \right\rceil \ell_{\{u,u'\}} \mu_{\{u,u'\}}^2\right) \leq \frac{1}{2}, \tag{5.32}$$

where C is defined in step 1 of the proof. This is equivalent to:

$$C\left(a\ell_{\{v,w\}}\mu_{\{v,w\}}^2 + d_{\{v,w\}} \sum_{\{u,u'\}\sim\{v,w\}} a \frac{b\ell_{\{u,u'\}}^2}{\ell_{\{v,w\}}} \mu_{\{u,u'\}}^2\right) \leq \frac{1}{8}$$
$$\text{if } \forall \{u,u'\} \sim \{v,w\}, \ell_{\{v,w\}} \leq b\ell_{\{u,u'\}},$$

where we bounded $\left\lceil b\frac{\ell_{\{v,w\}}}{\ell_{\{u,u'\}}}\right\rceil$ by $2\frac{b\ell_{\{v,w\}}}{\ell_{\{u,u'\}}}$ here. We here see that in this case, if

$$\mu^2_{\{v,w\}} = \frac{1}{2\ell_{\{v,w\}}} \times \min_{\{u,u'\}\sim\{v,w\}} \frac{\ell_{\{u,u'\}}}{\ell_{\{v,w\}}} \tag{5.33}$$

with $8a + 8d^2_{max}b \leq C^{-1}$, our inequality holds. However, our inequality on the ceil operator seems not to work in the general case. Let's take $\{u, u'\}$ a neighbor of $\{v, w\}$ such that $\ell_{\{v,w\}} > b\ell_{\{u,u'\}}$. As $\ell_{\{v,w\}} > b\ell_{\{u,u'\}}$, we have $\lceil\frac{b\ell_{\{u,u'\}}}{\ell_{\{v,w\}}}\rceil = 1$, leading to $a\lceil\frac{b\ell_{\{u,u'\}}}{\ell_{\{v,w\}}}\rceil\ell_{\{u,u'\}}\mu^2_{\{u,u'\}} = a\ell_{\{u,u'\}}\mu^2_{\{u,u'\}} \leq a \leq ab$. Hence, our result still holds.

Conclusion: We have our result for $C = \frac{1}{2a+8d^2_{max}ab}$ and a laplacian weighted with local communication constraints: $\mu^2_{\{v,w\}} = \frac{1}{2\ell_{\{v,w\}}} \times \min_{\{u,u'\}\sim\{v,w\}} \frac{\ell_{\{u,u'\}}}{\ell_{\{v,w\}}}$. The final rate thus depends on the smallest eigenvalue of the laplacian weighted by:

$$\frac{1}{2a + 8d^2_{max}ab}\frac{1}{L_{max}}\frac{1}{2\ell_{\{v,w\}}} \times \min_{\{u,u'\}\sim\{v,w\}} \frac{\ell_{\{u,u'\}}}{\ell_{\{v,w\}}}\,. \tag{5.34}$$

This ends the proof of Theorem 5.6. □

5.C.2. Adding stochasticity

We now prove the following result.

Theorem 5.7 (Adding Stochasticity)**.** *Assume that, for all $k \in \mathbb{N}$, there exists a $\mathcal{F}_{k+T-1}$-measurable event A_k, such that $\mathbb{P}(A_k|\mathcal{F}_k) \geq \frac{1}{2}$ almost surely, and that under A_k, Assumption 5.C.1 holds for all $k \leq \ell \leq k + T - 1$. Then, we have the following bound on $\Lambda_k(\mathbf{x})$:*

$$\mathbb{E}[\Lambda_k(\mathbf{x})] \leq \left(\frac{1}{4}(1-\gamma)^{T/3} + \frac{3}{4}\right)^{\lceil\frac{k}{2T}\rceil} \mathbb{E}[\Lambda_0]\,,$$

where γ is defined in Theorem 5.6.

Proof. Using the same arguments as in the proof of Theorem 5.6, we obtain:

$$\mathbb{E}[\Lambda_{t+1} - \Lambda_t|\mathcal{F}_t, A_t] \leq -\sigma\Lambda_t. \tag{5.35}$$

However, this is not enough to conclude. Under A^C_t, we only know that $\Lambda_{t+1} \leq \Lambda_t$ (our local coordinate gradient steps cannot increase distance to the optimum). Hence:

$$\mathbb{E}[\Lambda_{t+1}|\mathcal{F}_t] \leq (1 - \sigma\mathbb{I}_{A_t})\Lambda_t. \tag{5.36}$$

And then, by induction:

$$\mathbb{E}[\Lambda_t] \leq \mathbb{E}[P_t\Lambda_0], \text{ where } P_t = \prod_{s=0}^{t-1}(1 - \sigma\mathbb{I}_{A_s}). \tag{5.37}$$

However, no direct bound on P_t exists. The interdependencies on the events A_t make it impossible for an induction to prove a bound of the form $\leq (1-\sigma/2)^t$. However, the logarithm of the product seems easier to study:

$$\log(P_t) = \log(1 - \sigma)\sum_{s=0}^{t-1}\mathbb{I}_{A_s}, \tag{5.38}$$

giving us $\mathbb{E}\log(P_t) \leq \log(1 - \sigma)t/2$, as $\mathbb{P}(A_t) \geq 1/2$. We are thus going to make a study in

probability. For $t \in \mathbb{N}$, let $X_t = \frac{1}{T}\sum_{s=t}^{t+T-1} \mathbb{I}_{A_s}$. Using Markov-type inequalities conditionnaly on $\mathcal{F}_t$ gives:

$$\mathbb{P}(X_t \geq 1/3|\mathcal{F}_t) + 1/3\mathbb{P}(X_t \leq 1/3|\mathcal{F}_t) \geq \mathbb{E}[X_t|\mathcal{F}_t] \geq 1/2 \implies \mathbb{P}(X_t \geq 1/3|\mathcal{F}_t) \geq 1/4. \quad (5.39)$$

Thus, we have: $\mathbb{E}[\prod_{s=t}^{t+T-1}(1-\mathbb{I}_{A_s}\sigma)|\mathcal{F}_t] \leq \frac{1}{4}(1-\sigma)^{T/3} + \frac{3}{4}$. We then know how to control T consecutive factors of the product P_t. Skipping the next T terms, we have:

$$\mathbb{E}\left[\prod_{s=t}^{t+3T-1}(1-\mathbb{I}_{A_s}\sigma)\right] = \mathbb{E}\left[\prod_{s=t}^{t+T-1}(1-\mathbb{I}_{A_s}\sigma)\prod_{s=t+T}^{t+2T-1}(1-\mathbb{I}_{A_s}\sigma)\prod_{s=t+2T}^{t+3T-1}(1-\mathbb{I}_{A_s}\sigma)\right] \quad (5.40)$$

$$\leq \mathbb{E}\left[\prod_{s=t}^{t+T-1}(1-\mathbb{I}_{A_s}\sigma)\prod_{s=t+2T}^{t+3T-1}(1-\mathbb{I}_{A_s}\sigma)\right] \quad (5.41)$$

$$\leq \mathbb{E}\left[\prod_{s=t}^{t+T-1}(1-\mathbb{I}_{A_s}\sigma)\mathbb{E}^{\mathcal{F}_{t+2T}}\left\{\prod_{s=t+2T}^{t+3T-1}(1-\mathbb{I}_{A_s}\sigma)\right\}\right] \quad (5.42)$$

as in the last right hand side, the first big product is $\mathcal{F}_{t+2T}$-measurable (our asumption on the A_s states that they are $\mathcal{F}_{s+T-1}$-measurable). Then, using inequality $\mathbb{E}\left[\prod_{s=t}^{t+T-1}(1-\mathbb{I}_{A_s}\sigma)|\mathcal{F}_t\right] \leq \frac{1}{4}(1-\sigma)^{T/3} + \frac{3}{4}$ twice, with t and $t+2T$, we get:

$$\mathbb{E}\left[\prod_{s=t}^{t+3T-1}(1-\mathbb{I}_{A_s}\sigma)\right] \leq \mathbb{E}\left[\prod_{s=t}^{t+T-1}(1-\mathbb{I}_{A_s}\sigma)\left(\frac{1}{4}(1-\sigma)^{T/3}+\frac{3}{4}\right)\right]$$
$$\leq \left(\frac{1}{4}(1-\sigma)^{T/3}+\frac{3}{4}\right)^2.$$

Proceeding the same way by induction leads us to:

$$\mathbb{E}[P_t] \leq \left(\frac{1}{4}(1-\sigma)^{T/3}+\frac{3}{4}\right)^{\lfloor t/(2T)\rfloor}, \quad (5.43)$$

which is the desired bound. □

From the proof, we thus have the following corollary.

Corollary 5.C.1. *Assume that, for all $k \in \mathbb{N}$, there exists a $\mathcal{F}_{k+T-1}$-measurable event A_k, such that $\mathbb{P}(A_k|\mathcal{F}_k) \geq \frac{1}{2}$ almost surely, and that under A_k, Assumption 5.C.1 holds for all $k \leq \ell \leq k+T-1$. Then, we have the following bound on $\Lambda_k(\mathbf{x})$, for any $k \geq 0$:*

$$\mathbb{E}\left[\Lambda_{k+2T}(\mathbf{x})|\mathcal{F}_k\right] \leq \left(\frac{1}{4}(1-\gamma)^{T/3}+\frac{3}{4}\right)\mathbb{E}[\Lambda_k(\mathbf{x})|\mathcal{F}_k].$$

where γ is defined in Theorem 5.6.

5.C.3. Expliciting the constants in the loss networks model we consider

We now need to compute and tune the constants introduced in Theorem 5.6 for the assumptions of Theorem 5.7 to hold in our Loss Network model. We begin by the following lemma, inspired by queuing theory arguments, that upper bound the probability that an edge stays inactivated for a long period of time.

Note that we here come back to continuous time, to study the loss network model. What is important to keep in mind is that an edge cannot be occupied for a time longer than $\tau'_{\{v,w\}}$.

Lemma 5.C.2. *Let $\delta \in (0,1)$. For any $t_0 \geq 0$, $\{v,w\} \in E$, if the* Poisson *intensities are such that $p_{\{v,w\}} = \frac{1}{2\max(d_i,d_j)-1}(\tau'_{\{v,w\}})^{-1}$ and $\tau'_{max}(\{v,w\}) = \max_{\{u,u'\}\sim\{v,w\}} \tau'_{\{u,u'\}}$, let:*

$$\ell_{\{v,w\}} = \frac{\log(\delta^{-1})}{\log(1-(1-e^{-1})e^{-1})}(p^{-1}_{\{v,w\}} + \tau'_{max}(\{v,w\}))\,.$$

We have:

$$\mathbb{P}(\{v,w\} \text{ not activated in } [t_0, t_0+\ell_{\{v,w\}}]|\mathcal{F}_{t_0}) \leq \delta. \tag{5.44}$$

Proof of Lemma 5.C.2. Let $\{v,w\} \in E$ and $t_0 \geq 0$ fixed. We use tools from queuing theory [Tan95, $M/M/\infty/\infty$ queues] in order to compute the probability that edge $\{v,w\}$ is activable at a time t or not. More formally, we define a process $N_{\{v,w\}}(t)$ with values in $\mathbb{N}$, such that $N_{\{v,w\}}(t_0) = 1$ if $\{v,w\}$ non-available at time t_0 and 0 otherwise. Then, when an edge $\{u,u'\}$ such that $\{u,u'\} \sim \{v,w\}$ is activated, we make an increment of 1 on $N_{\{v,w\}}(t)$ (a *customer* arrives). This customer stays for a time $\tau'_{\{u,u'\}}$ and when he leaves, $N_{\{v,w\}}$ is decreased by 1. Thus $N_{\{v,w\}} \geq 0$ a.s., and if $N_{\{v,w\}} = 0$, then edge $\{v,w\}$ is available. For $t \geq \max_{\{u,u'\}\sim\{v,w\}} \tau'_{\{u,u'\}} + t_0$, $N_{\{v,w\}}(t)$ follows a Poisson law of parameter $\sum_{\{u,u'\}\sim\{v,w\}} p_{\{u,u'\}}\tau'_{\{u,u'\}}$. For any $t \geq \max_{\{u,u'\}\sim\{v,w\}} \tau'_{\{u,u'\}} + t_0$:

$$\mathbb{P}(\{v,w\} \text{ available at time } t|\mathcal{F}_{t_0}) \geq \mathbb{P}(N_i(t)=0) = \exp(-\sum_{\{u,u'\}\sim\{v,w\}} p_{\{u,u'\}}\tau'_{\{u,u'\}}).$$

That leads to taking $p_{\{u,u'\}} = \frac{1}{2}\frac{1}{\max(d_k,d_l)-1}(\tau'_{\{u,u'\}})^{-1}$ for all edges, in order to have

$$\mathbb{P}(\{v,w\} \text{ available at time } t|\mathcal{F}_{t_0}) \geq 1/e.$$

Then, $\mathbb{P}(\{v,w\} \text{ rings in } [t, t+p^{-1}_{\{v,w\}}]) = 1-e^{-1}$, giving:

$$\begin{aligned}
&\mathbb{P}(\{v,w\} \text{ activated in } [t_0, t_0+\tau'_{\max}(\{v,w\}) + p^{-1}_{\{v,w\}}]|\mathcal{F}_{t_0})\\
&\quad= \mathbb{P}(\{v,w\} \text{ rings in } [t, t+p^{-1}_{\{v,w\}}])\\
&\quad\times \mathbb{P}(\{v,w\} \text{ available at time } t|\mathcal{F}_{t_0},\\
&\qquad \{v,w\} \text{ rings at a time } t \in [t_0+\tau'_{\max}(\{v,w\}), t_0+\tau'_{\max}(\{v,w\}) + p^{-1}_{\{v,w\}}])\\
&\quad\geq (1-e^{-1})e^{-1},
\end{aligned}$$

where we use the memoriless property of exponential random variables. Take $k \in \mathbb{N}$ such that $(1-(1-e^{-1})e^{-1})^k \leq \delta$, leading to $k = \log(6|E|)/\log(1-(1-e^{-1})e^{-1})$. Let

$$\ell_{\{v,w\}} = k(p^{-1}_{\{v,w\}} + \tau'_{max}(\{v,w\})).$$

Then we have a.s.:

$$\mathbb{P}(\{v,w\} \text{ not activated in } [t_0, t_0+\ell_{\{v,w\}}]|\mathcal{F}_{t_0}) \leq \delta. \tag{5.45}$$

□

Let $t \in \mathbb{N}$ be fixed, and B_t be the event: "in the activations $t, t+1, ..., t+T-1$, all edges are activated". Let then $C_t(\{v,w\}, s)$ for $t \leq s < t+T$ be the event $\min(T_{\{v,w\}}(s), t+T-s, s-t) \leq a\ell_{\{v,w\}}$ and $D_t(\{u,u'\},\{v,w\},s)$ be the event $N(\{u,u'\},\{v,w\},s) \leq \lceil b\ell_{\{v,w\}}/\ell_{\{u,u'\}}\rceil$, where $N(\{u,u'\},\{v,w\},s)$ is the number of activations of $\{u,u'\}$ between two activations of $\{v,w\}$, around time s, where we only take into account the activations between activations t and $t+T-1$. Let then $A_t = B_t \cap (\cap_{\{u,u'\},\{v,w\}\in E, t\leq s<t+T} C_t(\{v,w\},s) \cap D_t(\{u,u'\},\{v,w\},s))$.

We want $\mathbb{P}(A_t) \geq 1/2$ for correct constants a, b, T and $\ell_{\{v,w\}}$ (that can differ from $\tau'_{\{v,w\}}$)

in order to apply Theorems 5.6 and 5.7. Note that this event is $\mathcal{F}_{t+T-1}$-measurable, as desired. We first study the length of time $\ell_{\{v,w\}}$ edge $\{v,w\}$ must wait in order to be activated with high probability (*high* meaning more that $1-\frac{1}{12|E|}$). This result is Lemma 5.C.2. Then, we use this length to determine the constants $T, a, b, \ell_{\{v,w\}}$ needed.

Lemma 5.C.3. *For any continuous time $t_0 \geq 0$, $\{v,w\} \in \mathcal{E}$, if $p_{\{v,w\}} = \frac{1}{2\max(d_i,d_j)-1}(\tau'_{\{v,w\}})^{-1}$ and $\tau'_{max}(\{v,w\}) = \max_{\{u,u'\}\sim\{v,w\}} \tau'_{\{u,u'\}}$, let $\ell_{\{v,w\}} = \frac{\log(6|E|)}{\log(1-(1-e^{-1})e^{-1})}(p^{-1}_{\{v,w\}}+\tau'_{max}(\{v,w\}))$. We have, almost surely:*

$$\mathbb{P}(\{v,w\} \text{ not activated in } [t_0, t_0+\ell_{\{v,w\}}]|\mathcal{F}_{t_0}) \leq \frac{1}{6|E|}. \tag{5.46}$$

Proof of Lemma 5.C.2. Let $\{v,w\} \in E$ and $t_0 \geq 0$ fixed. We use tools from queuing theory [Tan95] ($M/M/\infty/\infty$ queues) in order to compute the probability that edge $\{v,w\}$ is activable at a time t or not. More formally, we define a process $N_{\{v,w\}}(t)$ with values in $\mathbb{N}$, such that $N_{\{v,w\}}(t_0) = 1$ if $\{v,w\}$ non-available at time t_0 and 0 otherwise. Then, when an edge $\{u,u'\}, \{u,u'\} \sim \{v,w\}$ is activated, we make an increment of 1 on $N_{\{v,w\}}(t)$ (a *customer* arrives). This customer stays for a time $\tau'_{\{u,u'\}}$ and when he leaves we make $N_{\{v,w\}}$ decrease by 1. We have $N_{\{v,w\}} \geq 0$ a.s., and if $N_{\{v,w\}} = 0$, $\{v,w\}$ is available. For $t \geq \max_{\{u,u'\}\sim\{v,w\}} \tau'_{\{u,u'\}} + t_0$, $N_{\{v,w\}}(t)$ follows a Poisson law of parameter $\sum_{\{u,u'\}\sim\{v,w\}} p_{\{u,u'\}}\tau'_{\{u,u'\}}$. For any $t \geq \max_{\{u,u'\}\sim\{v,w\}} \tau'_{\{u,u'\}} + t_0$:

$$\mathbb{P}(\{v,w\} \text{ available at time } t|\mathcal{F}_{t_0}) \geq \mathbb{P}(N_i(t)=0) = \exp(-\sum_{\{u,u'\}\sim\{v,w\}} p_{\{u,u'\}}\tau'_{\{u,u'\}}). \tag{5.47}$$

That leads to taking $p_{\{u,u'\}} = \frac{1}{2}\frac{1}{\max(d_k,d_l)-1}(\tau'_{\{u,u'\}})^{-1}$ for all edges, in order to have

$$\mathbb{P}(\{v,w\} \text{ available at time } t|\mathcal{F}_{t_0}) \geq 1/e\,.$$

Then, $\mathbb{P}(\{v,w\} \text{ rings in } [t, t+p^{-1}_{\{v,w\}}]) = 1-e^{-1}$, giving:

$$\mathbb{P}(\{v,w\} \text{ activated in } [t_0, t_0+\tau'_{\max}(\{v,w\})+p^{-1}_{\{v,w\}}]|\mathcal{F}_{t_0}) = \mathbb{P}(\{v,w\} \text{ rings in } [t, t+p^{-1}_{\{v,w\}}]) \tag{5.48}$$

$$\times\, \mathbb{P}(\{v,w\} \text{ available at time } t|\mathcal{F}_{t_0}, \{v,w\} \text{ rings at a time} \tag{5.49}$$

$$t \in [t_0+\tau'_{\max}(\{v,w\}), t_0+\tau'_{\max}(\{v,w\})+p^{-1}_{\{v,w\}}]) \tag{5.50}$$

$$\geq (1-e^{-1})e^{-1}, \tag{5.51}$$

where we use the fact that exponential random variables have no memory. Take $k \in \mathbb{N}$ such that $(1-(1-e^{-1})e^{-1})^k \leq \frac{1}{6|E|}$, leading to $k \approx \log(6|E|)/\log(1-(1-e^{-1})e^{-1})$. Let $\ell_{\{v,w\}} = k(p^{-1}_{\{v,w\}} + \tau'_{max}(\{v,w\}))$. Then we have a.s.:

$$\mathbb{P}(\{v,w\} \text{ not activated in } [t_0, t_0+\ell_{\{v,w\}}]|\mathcal{F}_{t_0}) \leq \frac{1}{6|E|}. \tag{5.52}$$

□

Bounding T: A direct application of Lemma 5.C.2 leads, with $L = \max_{\{v,w\}} \ell_{\{v,w\}}$, to:

$$T = 2\sum_{\{v,w\}} \frac{L}{\tau'_{\{v,w\}}}. \tag{5.53}$$

Indeed, for all $\{v,w\}$, not being activated in activations $t, t+1, ..., t+T-1$ means not being activated for a continuous interval of time of length more than $\ell_{\{v,w\}}$. Hence:

$$\begin{aligned}
&\mathbb{P}(\exists(\{v,w\}) \in E : (\{v,w\}) \text{ not activated in } \{t, ..., t+T-1\}|\mathcal{F}_t) && (5.54)\\
&\leq \sum_{\{v,w\}\in E} \mathbb{P}((\{v,w\}) \text{ not activated in } \{t, ..., t+T-1\}|\mathcal{F}_t) && (5.55)\\
&\leq \sum_{\{v,w\}\in E} \mathbb{P}((\{v,w\}) \text{ not activated in } [t, t+\ell_{\{v,w\}}]|\mathcal{F}_t) && (5.56)\\
&\leq |E| \times \frac{1}{6|E|} && (5.57)\\
&= 1/6. && (5.58)
\end{aligned}$$

Bounding $T_{\{v,w\}}$: Applying Lemma 5.C.2 with $12|E|T$ instead of $6|E|$ leads to controlling all the inactivation lengths by a length $\ell'_{\{v,w\}}$, with a probability more than $1 - 1/(12|E|T)$. Let $\{v,w\} \in E$ and $s \in \mathbb{N}$, $t \leq s < t+T$. Let $\alpha > 0$ to tune later. Denote by $\delta_{\{v,w\}}(s)$ the (random) inactivation time of $\{v,w\}$, around iteration s. Note that conditionnaly on the inactivation period $\delta_{\{v,w\}}(s)$, $T_{\{v,w\}}(s)$ is dominated in law by a Poisson variable of parameter $I\delta_{\{v,w\}}(s)$, hence line (5.62):

$$\begin{aligned}
&\mathbb{P}(T_{\{v,w\}}(s) \geq \alpha\ell'_{\{v,w\}}|\mathcal{F}_t) && (5.59)\\
&\quad\leq \mathbb{P}(T_{\{v,w\}}(s) \geq \alpha\ell'_{\{v,w\}}|\mathcal{F}_t, \delta_{\{v,w\}} \leq \ell'_{\{v,w\}}) \times \mathbb{P}(\delta_{\{v,w\}} \leq \ell'_{\{v,w\}}) && (5.60)\\
&\qquad+ \mathbb{P}(\delta_{\{v,w\}} \geq \ell'_{\{v,w\}}) && (5.61)\\
&\quad\leq \mathbb{P}(Poisson(I\ell'_{\{v,w\}}) \geq \alpha\ell'_{\{v,w\}}) + \frac{1}{12|E|T} \quad (\text{where} \quad I = \sum_{\{v,w\}\in\mathcal{E}} p_{\{v,w\}}) && (5.62)\\
&\quad\leq \frac{1}{12|E|T} + \frac{1}{12|E|T} && (5.63)\\
&\quad= \frac{1}{6|E|T}, && (5.64)
\end{aligned}$$

for some $\alpha > 0$ big enough, to determine with the following large deviation inequality:

Lemma 5.C.4 (A Large Deviation Inequality on discrete Poisson variables.)**.** *Let $Z \sim Poisson(\lambda)$, for some $\lambda > 0$. Then, for all $u \geq 0$:*

$$\mathbb{P}(Z \geq u) \leq \exp(-u + \lambda(e-1)). \tag{5.65}$$

This large deviation leads to taking $\alpha = 2eI$ for (5.63) to be true. Finally, we get:

$$\mathbb{P}(T_{\{v,w\}}(s) \geq \alpha\ell'_{\{v,w\}}|\mathcal{F}_t) \leq \frac{1}{6|E|T}. \tag{5.66}$$

Bounding $N(\{u,u'\},\{v,w\},s)$: If $\delta_{\{v,w\}}(s) \leq \ell'_{\{v,w\}}$, this random variable is dominated by a Poisson variable of parameter $p_{\{u,u'\}}\ell'_{\{v,w\}}$. Hence, still with Lemma 5.C.4, with probability more than $1 - \frac{1}{12|E|^2T}$, we can bound $N(\{u,u'\},\{v,w\})$ by $e\log(12|E|^2T) + p_{\{u,u'\}}\ell_{\{v,w\}}(e-1) \leq 2ep_{\{u,u'\}}L_{\{v,w\}}$.

Explicit writing of the union bound on A_t^C: $A_t^C = B_t^C \cup (\cup_{\{u,u'\},\{v,w\}\in E, t\leq s<t+T} C_t(\{v,w\},s)^C \cup D_t(\{u,u'\},\{v,w\},s)^C) \in \mathcal{F}_{t+T-1}$. Thanks to the previous considerations, we have that $\mathbb{P}^{\mathcal{F}_t}(B_t^C) \leq 1/6$ with (5.58), $\mathbb{P}^{\mathcal{F}_t}(C_t(\{v,w\},s)^C) \leq \frac{1}{6|E|T}$ with (5.66) and $\mathbb{P}(D_t(\{u,u'\},\{v,w\},s)^C|\mathcal{F}_t) \leq \frac{1}{6|E|^2T}$, for the following constants and weights:

- $\tilde{\tau'}^{-1}_{\{v,w\}} = p_{\{v,w\}} = \min(\frac{1}{\tau'_{\max}(\{v,w\})}, \frac{1}{2(\max(d_i,d_j)-1)}\frac{1}{\tau'_{\{v,w\}}})$;
- $T = 2I \max_{\{v,w\}\in E} \tilde{\tau'}_{\{v,w\}} \frac{\log(6|E|)}{\log(1-(1-e^{-1})e^{-1})}$;
- $a = 2eI\frac{\log(6|E|T)}{\log(1-(1-e^{-1})e^{-1})}$;
- $b = 2e\frac{\log(6|E|T)}{\log(1-(1-e^{-1})e^{-1})}$.

The union bound is the following:

$$\begin{aligned}\mathbb{P}^{\mathcal{F}_t}(A_t^C) &\leq \mathbb{P}^{\mathcal{F}_t}(B_t^C) + \sum_{s,\{v,w\}} \mathbb{P}^{\mathcal{F}_t}(C_t(\{v,w\},s)^C) \\ &\quad + \sum_{s,\{v,w\}} \mathbb{P}^{\mathcal{F}_t}(\cup_{\{u,u'\}} D_t(\{u,u'\},\{v,w\},s)^C) \\ &\leq 1/6 + |E|T/(6|E|T) \times 2 \\ &\leq 1/2.\end{aligned}$$

The rate of convergence γ is then defined as the smallest non null eigenvalue of the laplacian of the graph, weighted by:

$$\nu_{\{v,w\}} = \frac{p_{\{v,w\}} \min_{\{u,u'\}\sim\{v,w\}} \frac{\tau'_{\{v,w\}}}{\tau_{\{u,u'\}}}}{8a(1+d^2b)} \tag{5.67}$$

$$= \frac{\min_{\{u,u'\}\sim\{v,w\}} p_{\{u,u'\}}}{c_1 \ln(6|\mathcal{E}|T)(1+d^2\ln(6|\mathcal{E}|T)^2)\sum_{\{u,u'\}\in\mathcal{E}} p_{\{u,u'\}}} \tag{5.68}$$

5.C.4. Concluding

What we have proved so far, is that for any $k \geq 0$, any $\mathbf{x} \in \mathbb{R}^{\mathcal{V}}$, we have:

$$\mathbb{E}\left[\Lambda_{k+2T}(\mathbf{x})|\mathcal{F}_k\right] \leq \left(\frac{1}{4}(1-\gamma)^{T/3} + \frac{3}{4}\right)\mathbb{E}[\Lambda_k(\mathbf{x})|\mathcal{F}_k]\,,$$

where γ is defined in Equation (5.67). Then, $\Lambda_{k+2T}(\mathbf{x}) \geq \frac{1}{2}\left\|W^{(0,k+2T)}(\mathbf{x}-\bar{\mathbf{x}})\right\|^2$ and $\Lambda_k(\mathbf{x}) \leq \frac{1}{2}\left\|W^{(0,k)}(\mathbf{x}-\bar{\mathbf{x}})\right\|^2$, so that applying this for $k=0$, almost surely conditionned on $\mathcal{F}_0$,

$$\mathbb{E}\left[\left\|W^{(0,2T)}(\mathbf{x}-\bar{\mathbf{x}})\right\|^2|\mathcal{F}_0\right] \leq \left(\frac{1}{4}(1-\gamma)^{T/3} + \frac{3}{4}\right)\mathbb{E}[\|\mathbf{x}-\bar{\mathbf{x}}\|^2|\mathcal{F}_0]\,,$$

Now, noticing that our analysis holds almost surely for any configuration $\mathcal{F}_0$, doing a time translation and starting from a configuration $\mathcal{F}_k$ for any k, we get that:

$$\mathbb{E}\left[\left\|W^{(k,k+2T)}(\mathbf{x}-\bar{\mathbf{x}})\right\|^2|\mathcal{F}_k\right] \leq \left(\frac{1}{4}(1-\gamma)^{T/3} + \frac{3}{4}\right)\mathbb{E}[\|\mathbf{x}-\bar{\mathbf{x}}\|^2|\mathcal{F}_k]\,,$$

so that Assumption 5.3.2 holds for $\rho = \frac{1}{4}(1-(1-\gamma)^{T/3}), k_\rho = 2T$, and hence $\frac{\rho}{k_\rho} = \mathcal{O}(\gamma)$, which leads to Theorem 5.5: γ is the eigengap of the graph, with weights of order $\tilde{\mathcal{O}}(\frac{\min_{\{u,u'\}\sim\{v,w\}} p_{\{u,u'\}}}{d^2\sum_{\{u,u'\}\in\mathcal{E}} p_{\{u,u'\}}})$.

5.D. Proof of Theorem 5.1: Convex-Lipchitz case

5.D.1. Homogeneous setting, Lipschitz (bounded gradients) and convex without sampling

Proof. Studying the virtual sequence, we expand:

$$\begin{aligned}\mathbb{E}\left[\left\|\hat{x}^{k+1}-x^\star\right\|^2\right] &= \mathbb{E}\left[\left\|\hat{x}^{k}-x^\star\right\|^2 - \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),\hat{x}^k-x^\star\rangle + \frac{\gamma^2}{n^2}\left\|\sum_{v\in\mathcal{I}_k} g_v^k\right\|^2\right]\\ &\le \mathbb{E}\Big[\left\|\hat{x}^{k}-x^\star\right\|^2 - \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),x_v^k-x^\star\rangle + \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),x_v^k-\bar{x}^k\rangle\\ &\quad + \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),\bar{x}^k-\hat{x}^k\rangle\Big]\\ &\quad + \frac{\gamma^2B^2|\mathcal{I}_k|^2}{n^2}\,,\end{aligned}$$

where we used the Lipschitz assumption, $\mathbb{E}\mathbf{g}_v^k = \nabla f_v(x_v^k)$ and boundness of gradients. Denote:

$$\begin{aligned}T_1 &= -\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),x_v^k-x^\star\rangle\\ T_2^k &= \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),x_v^k-\bar{x}^k\rangle\\ T_3 &= \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),\bar{x}^k-\hat{x}^k\rangle\,.\end{aligned}$$

Using convexity of f,

$$T_1 \le -\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\left(f(x_v^k)-f(x^\star)\right).$$

Using the Lipschitz assumption and Equation (5.10) that controls $\left\|\bar{x}^k-\hat{x}^k\right\|$, we bound T_3::

$$T_3 \le \frac{2\gamma^2B^2|\mathcal{I}_k|}{n}\,.$$

Using the Lipschitz assumption and our consensus bound from Equation (5.14), we bound T_2^k:

$$\begin{aligned}\sum_{k<K}T_2^k &\le \sum_{k<K}\frac{2\gamma B}{n}\sqrt{\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[\left\|x_v^k-\bar{x}^k\right\|^2\right]}\\ &\le \sum_{k<K}\frac{2\gamma B}{n}\sqrt{\mathbb{E}\left[\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\right]}\\ &\le \sum_{k<K}\frac{\gamma^2B^2}{n\bar{\rho}} + \frac{\bar{\rho}}{B}\mathbb{E}\left[\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\right]\\ &\le \frac{3\gamma^2B^2}{n\bar{\rho}}\sum_{k<K}|\mathcal{I}_k|\,.\end{aligned}$$

Consequently, denoting $\eta = \frac{\gamma}{n}$ and summing over $k < K$,

$$2\eta \sum_{k<K} \sum_{v\in\mathcal{I}_k} \mathbb{E}\left[f(x_v^k) - f(x^\star)\right] \le \mathbb{E}\left[\left\|\hat{x}^0 - x^\star\right\|^2\right] + \eta^2 B^2 \left(\mathbb{E}|\mathcal{I}_k| + 2n + 3n\bar{\rho}^{-1}\right) \sum_{k<K} |\mathcal{I}_k|$$
$$\le \mathbb{E}\left[\left\|\hat{x}^0 - x^\star\right\|^2\right] + \eta^2 B^2 \left(3n + 3n\bar{\rho}^{-1}\right) \sum_{k<K} |\mathcal{I}_k| \,.$$

Dividing by $2\eta \sum_{k<K} |\mathcal{I}_k|$,

$$\mathbb{E}\left[\frac{1}{\sum_{k<K}|\mathcal{I}_k|} \sum_{k<K} \sum_{v\in\mathcal{I}_k} f\left(x_v^k\right) - f(x_\star)\right] \le \frac{\mathbb{E}\left[\left\|\hat{x}^0 - x^\star\right\|^2\right]}{2\eta \sum_{k<K}|\mathcal{I}_k|} + \frac{\eta B^2}{2}(3n + 3n\bar{\rho}^{-1}) \,,$$

and

$$\mathbb{E}\left[\left\|\hat{x}^0 - x^\star\right\|^2\right] \le \left\|x^0 - x^\star\right\|^2 - 2\eta \sum_{v\in\mathcal{V}} \langle \nabla f(x^0), x^0 - x^\star\rangle + \eta^2 G^2/n$$
$$\le \left\|x^0 - x^\star\right\|^2 + \eta^2 B^2/K \,,$$

provided that $K \ge n$. Optimizing over η, we obtain that for $\eta = \sqrt{\frac{D^2}{2KB^2(3n+2n\bar{\rho}^{-1})}}$,

$$\mathbb{E}\left[f\left(\frac{1}{\sum_{k<K}|\mathcal{I}_k|} \sum_{k=0}^{K-1} \sum_{v\in\mathcal{I}_k} x_v^k\right) - f(x_\star)\right] \le 2\sqrt{\frac{2B^2D^2(3n + 2n\bar{\rho}^{-1})}{\sum_{k<K}|\mathcal{I}_k|}} \,.$$

□

5.D.2. Lipschitz (bounded gradients) and convex with sampling

Proof. Taking the proof just above, we still have

$$\mathbb{E}\left[\left\|\hat{x}^{k+1} - x^\star\right\|^2\right] \le \mathbb{E}\left[\left\|\hat{x}^k - x^\star\right\|^2 + T_1^k + T_2^k + T_3\right] + \frac{\gamma^2 B^2 |\mathcal{I}_k|^2}{n^2} \,.$$

We have, using convexity and then Lipschitzness:

$$T_1^k = -\frac{2\gamma}{n} \sum_{v\in\mathcal{I}_k} \langle \nabla f_v(x_v^k), x_v^k - x^\star\rangle$$
$$\le -\frac{2\gamma}{n} \sum_{v\in\mathcal{I}_k} p_v f_v(x_v^k) - f(x^\star)$$
$$= -\frac{2\gamma}{n} \sum_{v\in\mathcal{I}_k} p_v f_v(\bar{x}^k) - f(x^\star) + f_v(x_v^k) - f(\bar{x}^k)$$
$$\le -\frac{2\gamma}{n} \sum_{v\in\mathcal{I}_k} f_v(\bar{x}^k) - f(x^\star) - B\left\|x_v^k - \bar{x}^k\right\| \,,$$

so that

$$\mathbb{E}\left[T_1^k\right] \le -\frac{2\gamma\bar{p}}{n}(\mathbb{E}f(\bar{x}^k) - f(x^\star)) + \frac{2\gamma B}{n} \sum_{v\in\mathcal{V}} p_v \left\|x_v^k - \bar{x}^k\right\|$$
$$\le -\frac{2\gamma\bar{p}}{n}(\mathbb{E}f(\bar{x}^k) - f(x^\star)) + \frac{2\gamma B p_{\max}}{n}\sqrt{n}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\| \,.$$

We then have that:

$$\sum_{k<K} \frac{2\gamma B p_{\max}}{n} \sqrt{n} \left\| \mathbf{x}^k - \bar{\mathbf{x}}^k \right\| \leq 2\sqrt{2} \frac{\gamma^2 B^2 p_{\max}}{\sqrt{n}} \sqrt{K \sum_{k<K} |\mathcal{I}_k|}\,.$$

Then,

$$T_3 \leq \frac{2\gamma^2 B^2}{n}\,.$$

We handle the consensus term differently. For some $\alpha > 0$ to be fix later, and taking the expectation conditionnally on $\mathbf{x}^k$,

$$\begin{aligned}
\sum_{k<K} \mathbb{E}\left[T_2^k\right] &\leq \sum_{k<K} \sum_{v\in\mathcal{I}_k} \mathbb{E}\left[\frac{\gamma^2}{n\alpha}\left\|\nabla f(x_v^k)\right\|^2 + \frac{\alpha}{n}\left\|x_v^k - \bar{x}^k\right\|^2\right] \\
&\leq \sum_{k<K} \frac{\gamma^2 B^2 |\mathcal{I}_k|}{\alpha n} + \frac{\alpha}{n} \sum_{v\in\mathcal{V}} p_v \left\|x_v^k - \bar{x}^k\right\|^2 \\
&\leq \frac{\gamma^2 B^2}{\alpha n} \sum_{k<K} |\mathcal{I}_k| + \frac{\alpha p_{\max}}{n} \sum_{k<K} \mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 \\
&\leq \left(\frac{\gamma^2 B^2}{\alpha n} + \frac{\alpha p_{\max}}{n} \frac{2\gamma^2 B^2}{\bar{\rho}^2}\right) \sum_{k<K} |\mathcal{I}_k|\,.
\end{aligned}$$

We set $\alpha = 1/\sqrt{p_{\max}\bar{\rho}^{-2}}$, so that:

$$\sum_{k<K} \mathbb{E}\left[T_2^k\right] \leq 2\frac{\gamma^2 B^2}{n^2} \times \sqrt{p_{\max}} n \bar{\rho}^{-1} \times \sum_{k<K} |\mathcal{I}_k|\,.$$

The rest of the proof then follows as before, and we obtain

$$\begin{aligned}
&\mathbb{E}\left[f\left(\frac{1}{\sum_{k<K}|\mathcal{I}_k|} \sum_{k=0}^{K-1} \sum_{v\in\mathcal{I}_k} x_v^k\right) - f(x^\star)\right] \\
&\quad = \mathcal{O}\left(\sqrt{\frac{B^2 D^2}{\sum_{k<K}|\mathcal{I}_k|}\left(n + (p_{\max})^{1/2} n\bar{\rho}^{-1} + n^{3/2} p_{\max} \sqrt{\frac{K}{\sum_{k<K}|\mathcal{I}_k|}}\right)}\right)\,.
\end{aligned}$$

To conclude, we notice that $\frac{nK}{\sum_{k<K}|\mathcal{I}_k|}$ is of order $1/\bar{p}$ where $\bar{p} = \frac{1}{n}\sum_{v\in\mathcal{V}}$. □

5.E. Proof of Theorem 5.2: smooth-Lipschitz-convex rates

5.E.1. Smooth-Lipschitz-convex rates without sampling, homogeneous case

Proof. As before, we have:

$$\mathbb{E}\left[\left\|\hat{x}^{k+1} - x^\star\right\|^2\right] \leq \mathbb{E}\left[\left\|\hat{x}^k - x^\star\right\|^2 + T_1 + T_2^k + T_3\right] + \frac{\gamma^2\sigma^2|\mathcal{I}_k| + \gamma^2\mathbb{E}\left\|\sum_{v\in\mathcal{I}_k}\nabla f_v(x_v^k)\right\|^2}{n^2}\,,$$

with

$$T_1 = -\frac{2\gamma}{n} \sum_{v\in\mathcal{I}_k} \langle \nabla f_v(x_v^k), x_v^k - x^\star\rangle$$

$$T_2^k = \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle \nabla f_v(x_v^k), x_v^k - \bar{x}^k\rangle$$
$$T_3 = \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle \nabla f_v(x_v^k), \bar{x}^k - \hat{x}^k\rangle\,.$$

First, using convexity of $f_v \equiv f$,

$$T_1 \le -\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}(f_v(x_v^k) - f_v(x^\star)) = -\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}(f(x_v^k) - f(x^\star))\,.$$

Using Assumption 5.A.1 we have, where $C>0$ can be arbitrary:

$$\begin{aligned}\mathbb{E}\left[T_2^k\right] &\le \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[\left\|\nabla f(x_v^k)\right\|\left\|x_v^k - \bar{x}^k\right\|\right]\\ &\le \frac{C\gamma}{n}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[\left\|\nabla f(x_v^k)\right\|^2\right] + \frac{\gamma}{Cn}\mathbb{E}\left[\sum_{v\in\mathcal{I}_k}\left\|x_{v_k}^k - \bar{x}^k\right\|^2\right]\\ &\le \frac{2LC\gamma}{n}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[(f(x_v^k) - f(x^\star))\right] + \frac{\gamma}{Cn}\mathbb{E}\left[\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2\right].\end{aligned}$$

We also have:

$$\begin{aligned}T_3 &\le \frac{\gamma}{n}\Big(C\sum_{v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2 + \frac{1}{C}\left\|\bar{x}^k - \hat{x}^k\right\|^2\Big)\\ &\le \frac{\gamma}{n}\Big(2LC\sum_{v\in\mathcal{I}_k}(f(x_v^k) - f(x^\star)) + \frac{\gamma^2B^2}{C}|\mathcal{I}_k|\Big)\,.\end{aligned}$$

Thus,

$$\begin{aligned}&\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k) - f(x^\star))\\ &\le -\mathbb{E}\left[\left\|\hat{x}^{k+1} - x^\star\right\|^2\right] + \mathbb{E}\left[\left\|\hat{x}^k - x^\star\right\|^2\right] + \frac{\gamma^2\sigma^2|\mathcal{I}_k|}{n^2} + \frac{2\gamma^2L|\mathcal{I}_k|}{n^2}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k) - f(x^\star))\\ &+ \frac{2LC\gamma}{n}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[(f(x_v^k) - f(x^\star))\right] + \frac{\gamma}{Cn}\mathbb{E}\left[\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2\right]\\ &+ \frac{\gamma}{n}\Big(2LC\sum_{v\in\mathcal{I}_k}(f(x_v^k) - f(x^\star)) + \frac{\gamma^2B^2}{C}|\mathcal{I}_k|\Big)\,.\end{aligned}$$

Summing over $k<K$ and using Lemma 5.B.2, we obtain:

$$\begin{aligned}\frac{2\gamma}{n}\sum_{k<K}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k) - f(x^\star)) &\le \mathbb{E}\left[\left\|\hat{x}^0 - x^\star\right\|^2\right] + \frac{\gamma^2\sigma^2}{n^2}\sum_{k<K}|\mathcal{I}_k| + \frac{2\gamma L}{n}(2C+\gamma)\sum_{k<K}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k) - f(x^\star))\\ &+ \sum_{k<K}\frac{\gamma}{Cn}\mathbb{E}\left[\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2\right] + \frac{\gamma^3B^2}{Cn}\sum_{k<K}|\mathcal{I}_k|\\ &\le \mathbb{E}\left[\left\|\hat{x}^0 - x^\star\right\|^2\right] + \frac{\gamma^2\sigma^2}{n^2}\Big(1 + \frac{2\gamma\bar{\rho}^{-1}n}{C}\Big)\sum_{k<K}|\mathcal{I}_k| + \sum_{k<K}\frac{\gamma}{Cn}\mathbb{E}\left[\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2\right]\end{aligned}$$

$$+\frac{\gamma^3 B^2}{Cn}\sum_{k<K}|\mathcal{I}_k| + \frac{2\gamma L}{n}\big(2C+\gamma+\frac{4\gamma^2}{\bar\rho^2}\big)\sum_{k<K}\sum_{v\in\mathcal{I}_k}\left(\mathbb{E}f(x_v^k)-f(x^\star)\right).$$

Hence, provided that $2C+\gamma+\frac{4\gamma^2}{\bar\rho^2}\leq\frac{1}{2L}$, which is verified for $C=\frac{1}{8L}$ and $\gamma\leq\frac{1}{4L}\times\frac{1}{1+2\bar\rho^{-1}}$, we have:

$$\frac{\gamma}{n}\sum_{k<K}\sum_{v\in\mathcal{I}_k}\left(\mathbb{E}f(x_v^k)-f(x^\star)\right)\leq\mathbb{E}\left[\left\|\hat x^0-x^\star\right\|^2\right]+\frac{\gamma^2\sigma^2}{n^2}\big(1+16L\gamma\bar\rho^{-1}n\big)\sum_{k<K}|\mathcal{I}_k|+\frac{8L\gamma^3B^2}{n}\sum_{k<K}|\mathcal{I}_k|,$$

leading to, for $\eta=\gamma/n$:

$$\mathbb{E}\left[f\left(\frac{1}{\sum_{k<K}|\mathcal{I}_k|}\sum_{k=0}^{K-1}\sum_{v\in\mathcal{I}_k}x_v^k\right)-f(x^\star)\right]\leq\frac{\mathbb{E}\left[\left\|\hat x^0-x^\star\right\|^2\right]}{\eta\sum_{k<K}|\mathcal{I}_k|}+\eta\sigma^2+\eta^2\left(16L\sigma^2n^2\bar\rho^{-1}+8LB^2n^2\right).$$

Optimizing over $\eta\leq\frac{1}{4L}\times\frac{1}{n(1+2\bar\rho^{-1})}$, we thus obtain that:

$$\begin{aligned}&\mathbb{E}\left[f\left(\frac{1}{\sum_{k<K}|\mathcal{I}_k|}\sum_{k=0}^{K-1}\sum_{v\in\mathcal{I}_k}x_v^k\right)-f(x^\star)\right]\\&=\mathcal{O}\left(\frac{LD^2n\bar\rho^{-1}}{\sum_{k<K}|\mathcal{I}_k|}+\sqrt{\frac{D\sigma^2}{\sum_{k<K}|\mathcal{I}_k|}}+\left[\frac{D^2\sqrt{LB^2n^2+L\sigma^2n^2\bar\rho^{-1}}}{\sum_{k<K}|\mathcal{I}_k|}\right]^{2/3}\right).\end{aligned}$$

□

5.E.2. Smooth-Lipschitz-convex rates with sampling, heterogeneous case

Proof. We have:

$$\mathbb{E}\left[\left\|\hat x^{k+1}-x^\star\right\|^2\right]\leq\mathbb{E}\left[\left\|\hat x^k-x^\star\right\|^2-\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),\hat x^k-x^\star\rangle\right]+\frac{\gamma^2\sigma^2|\mathcal{I}_k|+\gamma^2\mathbb{E}\left\|\sum_{v\in\mathcal{I}_k}\nabla f_v(x_v^k)\right\|^2}{n^2},$$

and we will handle the middle term differently than before. Using $-\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),\hat x^k-x^\star\rangle=-\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),x_v^k-x^\star\rangle-\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),\hat x^k-x_v^k\rangle$ and then convexity for the first term and smoothness for the second, we obtain:

$$\begin{aligned}&-\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),\hat x^k-x^\star\rangle\\&\leq-\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\left(f_v(x_v^k)-f_v(x^\star)-\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}f_v(\hat x^k)-f_v(x_v^k)-\frac{L}{2}\left\|x_v^k-\hat x^k\right\|^2\right)\\&=-\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}f_v(\hat x^k)-f_v(x^\star)+\frac{\gamma L}{n}\sum_{v\in\mathcal{I}_k}\left\|x_v^k-\hat x^k\right\|^2.\end{aligned}$$

Taking the expectation wrt $\mathcal{I}_k$:

$$\mathbb{E}\left[-\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle\nabla f_v(x_v^k),\hat x^k-x^\star\rangle\right]\leq-\frac{2\gamma}{n}\sum_{v\in\mathcal{V}}p_v\big(f_v(\hat x^k)-f_v(x^\star)\big)+\frac{\gamma L}{n}\sum_{v\in\mathcal{V}}p_v\left\|x_v^k-\hat x^k\right\|^2$$

$$\leq -\frac{2\gamma n\bar{p}}{n}\big(f(\hat{x}^k)-f(x^\star)\big)+\frac{2\gamma L p_{\max}}{n}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+\frac{\gamma L}{n}\sum_{v\in\mathcal{V}}p_v\left\|\hat{x}^k-\bar{x}^k\right\|^2$$
$$\leq -\frac{2\gamma n\bar{p}}{n}\big(f(\hat{x}^k)-f(x^\star)\big)+\frac{2\gamma L p_{\max}}{n}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+2\gamma L\bar{p}\left\|\hat{x}^k-\bar{x}^k\right\|^2.$$

Then, for the variance term, we need to bound $\mathbb{E}\left\|\sum_{v\in\mathcal{I}_k}\nabla f_v(x_v^k)\right\|^2$. For any $(z_v)_{v\in\mathcal{V}}$, we have $\mathbb{E}\left[\left\|\sum_{v\in\mathcal{I}_k}z_v\right\|^2\right]=\mathbb{E}\left[\sum_{v,v'\in\mathcal{V}}\mathbb{1}_{v\in\mathcal{V}}\mathbb{1}_{v'\in\mathcal{V}}\langle z_v,z_{v'}\rangle\right]=\sum_{v\neq v'\in\mathcal{V}}\mathbb{1}_{v\in\mathcal{V}}p_vp_{v'}\langle z_v,z_{v'}\rangle+\sum_{v\in\mathcal{V}}p_v\|z_v\|^2\leq\sum_{v\in\mathcal{V}}p_v\|z_v\|^2+\left\|\sum_{v\in\mathcal{V}}p_vz_v\right\|^2$. And finally, using convexity of the squared norm, $\left\|\sum_{v\in\mathcal{V}}p_vz_v\right\|^2\leq n\bar{p}\sum_{v\in\mathcal{V}}p_v\|z_v\|^2$. Hence, we have

$$\mathbb{E}_{\mathcal{I}_k}\left\|\sum_{v\in\mathcal{I}_k}\nabla f_v(x_v^k)\right\|^2\leq\sum_{v\in\mathcal{V}}p_v\left\|\nabla f_v(x_v^k)\right\|^2+\left\|\sum_{v\in\mathcal{V}}p_v\nabla f_v(x_v^k)\right\|^2.$$

Thus, plugging this in the first inequality,

$$\begin{aligned}\frac{2\gamma n\bar{p}}{n}\big(\mathbb{E}f(\hat{x}^k)-f(x^\star)\big)&\leq\mathbb{E}\left[\left\|\hat{\mathbf{x}}^k-\mathbf{x}^\star\right\|^2-\left\|\hat{\mathbf{x}}^{k+1}-\mathbf{x}^\star\right\|^2\right]+\mathbb{E}\left[\frac{\gamma^2\sigma^2|\mathcal{I}_k|+\gamma^2\mathbb{E}\left\|\sum_{v\in\mathcal{I}_k}\nabla f_v(x_v^k)\right\|^2}{n^2}\right]\\&\quad+\mathbb{E}\left[\frac{2\gamma Lp_{\max}}{n}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+2\gamma L\bar{p}\left\|\hat{x}^k-\bar{x}^k\right\|^2\right]\\&=\mathbb{E}\left[\left\|\hat{\mathbf{x}}^k-\mathbf{x}^\star\right\|^2-\left\|\hat{\mathbf{x}}^{k+1}-\mathbf{x}^\star\right\|^2\right]\\&\quad+\frac{\gamma^2\sigma^2n\bar{p}+\gamma^2\sum_{v\in\mathcal{V}}p_v\mathbb{E}\left\|\nabla f_v(x_v^k)\right\|^2+\gamma^2\left\|\sum_{v\in\mathcal{V}}p_v\nabla f_v(x_v^k)\right\|^2}{n^2}\\&\quad+\mathbb{E}\left[\frac{2\gamma Lp_{\max}}{n}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+2\gamma L\bar{p}\left\|\hat{x}^k-\bar{x}^k\right\|^2\right].\end{aligned}$$

Then, using smoothness, we have that $f(\bar{x}^k)-f(x^\star)\leq f(\hat{x}^k)-f(x^\star)+\langle\nabla f(\hat{x}^k),\hat{x}^k-\bar{x}^k\rangle+\frac{L}{2}\left\|\bar{x}^k-\hat{x}^k\right\|\leq 2(f(\hat{x}^k)-f(x^\star))+2L\left\|\bar{x}^k-\hat{x}^k\right\|^2$, leading to:

$$\begin{aligned}\frac{2\gamma n\bar{p}}{n}\big(\mathbb{E}f(\bar{x}^k)-f(x^\star)\big)&\leq\frac{4\gamma n\bar{p}}{n}\big(\mathbb{E}f(\hat{x}^k)-f(x^\star)\big)+\frac{4L\gamma n\bar{p}}{n}\left\|\bar{x}^k-\hat{x}^k\right\|^2\\&\leq 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^k-\mathbf{x}^\star\right\|^2-\left\|\hat{\mathbf{x}}^{k+1}-\mathbf{x}^\star\right\|^2\right]\\&\quad+\frac{2\gamma^2\sigma^2n\bar{p}+2\gamma^2\left(\sum_{v\in\mathcal{V}}p_v\mathbb{E}\left\|\nabla f_v(x_v^k)\right\|^2+\left\|\sum_{v\in\mathcal{V}}p_v\nabla f_v(x_v^k)\right\|^2\right)}{n^2}\\&\quad+\mathbb{E}\left[\frac{4\gamma Lp_{\max}}{n}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+8\gamma L\bar{p}\left\|\hat{x}^k-\bar{x}^k\right\|^2\right].\end{aligned}$$

We have $\left\|\hat{x}^k-\bar{x}^k\right\|^2\leq\gamma^2B^2$. Now,

$$\begin{aligned}\sum_{v\in\mathcal{V}}p_v\left\|\nabla f_v(x_v^k)\right\|^2&\leq 2\sum_{v\in\mathcal{V}}p_v\left\|\nabla f_v(x_v^k)-\nabla f_v(\bar{x}^k)\right\|^2+p_v\left\|\nabla f_v(\bar{x}^k)\right\|^2\\&\leq 2L^2p_{\max}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+2\sum_{v\in\mathcal{V}}p_v\left\|\nabla f_v(\bar{x}^k)\right\|^2\end{aligned}$$

$$\leq 2L^2 p_{max}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 + 2n\bar{p}\left\|\nabla f(\bar{x}^k)\right\|^2 + 2n\bar{p}\zeta^2\,.$$

Then,

$$\begin{aligned}\left\|\sum_{v\in\mathcal{V}} p_v \nabla f_v(x_v^k)\right\|^2 &\leq 2\left\|\sum_{v\in\mathcal{V}} p_v \nabla f_v(\bar{x}^k)\right\|^2 + 2\left\|\sum_{v\in\mathcal{V}} p_v(\nabla f_v(x_v^k) - \nabla f_v(\bar{x}^k))\right\|^2\\ &\leq 2(n\bar{p})^2\left\|\nabla f(\bar{x}^k)\right\|^2 + 2(n\bar{p})\sum_{v\in\mathcal{V}} p_v\left\|(\nabla f_v(x_v^k) - \nabla f_v(\bar{x}^k))\right\|^2\\ &\leq 2(n\bar{p})^2\left\|\nabla f(\bar{x}^k)\right\|^2 + 2(n\bar{p})\sum_{v\in\mathcal{V}} p_v L^2\left\|x_v^k - \bar{x}^k\right\|^2\\ &\leq 2(n\bar{p})^2\left\|\nabla f(\bar{x}^k)\right\|^2 + 2(n\bar{p})p_{\max}L^2\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2\end{aligned}$$

Thus, this leads to:

$$\begin{aligned}&\frac{2\gamma n\bar{p}}{n}\left(\mathbb{E}f(\bar{x}^k) - f(x^\star)\right)\\ &\quad\leq 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^k - \mathbf{x}^\star\right\|^2 - \left\|\hat{\mathbf{x}}^{k+1} - \mathbf{x}^\star\right\|^2\right] + \frac{2\gamma^2(\sigma^2 + 2\zeta^2)n\bar{p} + 8\gamma^2 n^2\bar{p}^2\left\|\nabla f(\bar{x}^k)\right\|^2}{n^2}\\ &\qquad + \mathbb{E}\left[\left(\frac{4\gamma L p_{\max}}{n} + \frac{2\gamma^2 L^2 p_{\max}(1+n\bar{p})}{n}\right)\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 + 8\gamma L\bar{p}\left\|\hat{x}^k - \bar{x}^k\right\|^2\right].\end{aligned}$$

We now use the following lemma.

Lemma 5.E.1. *For stepsizes $\gamma \leq \frac{\bar{\rho}}{4L\sqrt{p_{\max}}}$, we have:*

$$\sum_{k<K}\mathbb{E}\left[\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2\right] \leq 4\gamma^2\sigma^2\bar{\rho}^{-1}n\bar{p}K + 8\gamma^2\bar{\rho}^{-2}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2 + 16\gamma^2\bar{\rho}^{-2}n\bar{p}\sum_{k<K}\left(\left\|\nabla f(\bar{x}^k)\right\|^2 + \zeta^2\right).$$

Proof of the lemma. Denoting $C_K = \sum_{k<K}\mathbb{E}\left[\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2\right]$ and using Lemma 5.B.2, we have

$$\begin{aligned}C_k &\leq 2\gamma^2\sigma^2\bar{\rho}^{-1}\sum_{k<K}|\mathcal{I}_k| + \frac{4\gamma^2}{\bar{\rho}^2}\sum_{k<K}\mathbb{E}\left[\sum_{v\in\mathcal{I}_k}\left\|\nabla f_v(x_v^{k-\tau(k,v)})\right\|^2\right]\\ &\leq 2\gamma^2\sigma^2\bar{\rho}^{-1}n\bar{p}K + 8\gamma^2\bar{\rho}^{-2}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2 + \frac{4\gamma^2}{\bar{\rho}^2}\sum_{k<K}\sum_{v\in\mathcal{V}}p_v\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\right],\end{aligned}$$

using $\sum_{k<K}\sum_{v\in\mathcal{I}_k}\left\|\nabla f_v(x_v^{k-\tau(k,v)})\right\|^2 \leq \sum_{k<K}\sum_{v\in\mathcal{I}_k}\left\|\nabla f_v(x_v^k)\right\|^2 + \sum_{v\in\mathcal{V}}\left\|\nabla f_v(x_v^0)\right\|^2$. Then, $\sum_{v\in\mathcal{V}}p_v\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\right] \leq 2\sum_{v\in\mathcal{V}}p_v\mathbb{E}\left[\left\|\nabla f_v(\bar{x}^k)\right\|^2\right] + 2\sum_{v\in\mathcal{V}}p_v\mathbb{E}\left[\left\|\nabla f_v(\bar{x}^k) - \nabla f_v(x_v^k)\right\|^2\right] \leq 2n\bar{p}\zeta^2 + 2n\bar{p}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 + 2L^2p_{\max}\mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2$, which leads to:

$$\begin{aligned}C_K &\leq 2\gamma^2\sigma^2\bar{\rho}^{-1}n\bar{p}K + 4\gamma^2\bar{\rho}^{-2}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2 + 8\gamma^2\bar{\rho}^{-2}n\bar{p}\sum_{k<K}\left(\left\|\nabla f(\bar{x}^k)\right\|^2 + \zeta^2\right)\\ &\quad + 8\gamma^2L^2p_{\max}\bar{\rho}^{-2}C_K\,,\end{aligned}$$

leading to the desired result for $\gamma \leq \frac{\bar{\rho}}{4L\sqrt{p_{\max}}}$. □

Using Lemma 5.B.1 and Lemma 5.E.1, we thus have:

$$\begin{aligned}
\frac{2\gamma n\bar{p}}{n}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k)-f(x^\star)\right) &\leq 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0-\mathbf{x}^\star\right\|^2\right]+\frac{2\gamma^2(\sigma^2+2\zeta^2)\bar{p}K}{n}+4\gamma^2\bar{p}\sum_{k<K}\mathbb{E}\left[\left\|\nabla f(\bar{x}^k)\right\|^2\right]\\
&\quad+\mathbb{E}\left[\left(\frac{4\gamma Lp_{\max}}{n}+\frac{2\gamma^2L^2p_{\max}}{n}\right)\sum_{k<K}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\right]+8\gamma^3LB^2\bar{p}K\\
&\leq 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0-\mathbf{x}^\star\right\|^2\right]+\frac{2\gamma^2(\sigma^2+2\zeta^2)\bar{p}K}{n}+4\gamma^2\bar{p}\sum_{k<K}\mathbb{E}\left[\left\|\nabla f(\bar{x}^k)\right\|^2\right]++8\gamma^3LB^2\bar{p}K\\
&\quad+\frac{6\gamma Lp_{\max}}{n}\left[4\gamma^2\sigma^2\bar{\rho}^{-1}n\bar{p}K+8\gamma^2\bar{\rho}^{-2}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2+16\gamma^2\bar{\rho}^{-2}n\bar{p}\sum_{k<K}\left(\left\|\nabla f(\bar{x}^k)\right\|^2+\zeta^2\right)\right]\\
&=2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0-\mathbf{x}^\star\right\|^2\right]+\left(8\gamma^2L\bar{p}+96\gamma^3L^2p_{\max}\bar{p}\bar{\rho}^{-2}\right)\sum_{k<K}\mathbb{E}\left[f(\bar{x}^k)-f(x^\star)\right]+\frac{2\gamma^2(\sigma^2+2\zeta^2)\bar{p}K}{n}\\
&\quad+\gamma^3K\left(8LB^2\bar{p}+24L\sigma^2p_{\max}\bar{p}\bar{\rho}^{-1}+96L\zeta^2p_{\max}\bar{p}\bar{\rho}^{-2}\right)+\frac{48\gamma^2Lp_{\max}\bar{\rho}^{-2}}{n}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2.
\end{aligned}$$

Hence, for stepsizes satisfying $8\gamma L\bar{p}+96\gamma^2L^2p_{\max}\bar{p}\bar{\rho}^{-2}\leq\bar{p}$, which is verified for $\gamma\leq\min\left(\frac{1}{16L},\frac{\bar{\rho}}{14L\sqrt{p_{\max}}}\right)$, we obtain:

$$\begin{aligned}
\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k)-f(x^\star)\right)&\leq\frac{2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0-\mathbf{x}^\star\right\|^2\right]}{\gamma\bar{p}}+\frac{2\gamma(\sigma^2+2\zeta^2)K}{n}+\gamma^2K\left(8LB^2+24L\sigma^2p_{\max}\bar{\rho}^{-1}+96L\zeta^2p_{\max}\bar{\rho}^{-2}\right)\\
&\quad+\frac{48\gamma Lp_{\max}\bar{\rho}^{-2}}{n\bar{p}}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2.
\end{aligned}$$

Optimizing over $\gamma\leq\min\left(\frac{1}{16L},\frac{\bar{\rho}}{14L\sqrt{p_{\max}}},\frac{\bar{p}}{L}\right)$, this leads to:

$$\begin{aligned}
\frac{1}{K}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k)-f(x^\star)\right)=\mathcal{O}\Bigg(&\frac{LD^2\left(\frac{1}{\bar{p}}+\sqrt{\frac{p_{\max}}{\bar{p}^2}}\bar{\rho}^{-1}\right)}{K}+\left[\frac{D^2\sqrt{LB^2+L\sigma^2p_{\max}\bar{\rho}^{-1}+L\zeta p_{\max}\bar{\rho}^{-2}}}{\bar{p}K}\right]^{\frac{2}{3}}\\
&+\sqrt{\frac{D^2(\sigma^2+\zeta^2)}{n\bar{p}K}}+\frac{\bar{\rho}^{-1}\frac{p_{\max}}{\bar{p}}}{K}\frac{1}{n}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2\Bigg).
\end{aligned}$$

□

5.F. Proof of Theorem 5.3: smooth-convex case

5.F.1. Homogeneous without sampling

Proof. As before, we have:

$$\mathbb{E}\left[\left\|\hat{x}^{k+1}-x^\star\right\|^2\right]\leq\mathbb{E}\left[\left\|\hat{x}^k-x^\star\right\|^2+T_1+T_2^k+T_3\right]+\frac{\gamma^2\sigma^2|\mathcal{I}_k|+\gamma^2\mathbb{E}\left\|\sum_{v\in\mathcal{I}_k}\nabla f_v(x_v^k)\right\|^2}{n^2},$$

with

$$T_1 = -\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle \nabla f_v(x_v^k), x_v^k - x^\star\rangle$$
$$T_2^k = \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle \nabla f_v(x_v^k), x_v^k - \bar{x}^k\rangle$$
$$T_3 = \frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}\langle \nabla f_v(x_v^k), \bar{x}^k - \hat{x}^k\rangle\,,$$

We will bound T_1, T_2 as in the proof with the Lipschitz assumption. For the term T_3, using convexity and Lemma 5.B.1:

$$\begin{aligned}\mathbb{E}T_3 &\le \frac{\gamma}{n}\Big(C\sum_{v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2 + \frac{1}{C}\mathbb{E}\left\|\bar{x}^k - \hat{x}^k\right\|^2\Big)\\ &\le \frac{\gamma}{n}\Big(2LC\sum_{v\in\mathcal{I}_k}(f(x_v^k) - f(x^\star)) + \frac{2\gamma^2}{Cn}|\mathcal{I}_k|(\sigma^2 + \sum_{v\in\mathcal{V}}\left\|\nabla f(x_v^{k-\tau(v,k)})\right\|^2)\Big)\\ &\le \frac{2\gamma^2}{Cn^2}|\mathcal{I}_k|\sigma^2 + \frac{\gamma}{n}\Big(2LC\sum_{v\in\mathcal{I}_k}(f(x_v^k) - f(x^\star)) + \frac{2\gamma^2}{Cn}|\mathcal{I}_k|\sum_{v\in\mathcal{V}}\left\|\nabla f(x_v^{k-\tau(v,k)})\right\|^2\Big)\,.\end{aligned}$$

for $\gamma \le 1/(nL)$. Then,

$$\begin{aligned}\sum_{k<K}|\mathcal{I}_k|\sum_{v\in\mathcal{V}}\left\|\nabla f(x_v^{k-\tau(v,k)})\right\|^2 &\le \sum_{v\in\mathcal{V}}\sum_{k<K:v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2\sum_{\ell=k}^{\mathrm{next}(v,k+1)-1}|\mathcal{I}_\ell|\\ &\le \tau_{\max}\sum_{v\in\mathcal{V}}\sum_{k<K:v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2\\ &\le 2L\tau_{\max}\sum_{v\in\mathcal{V}}\sum_{k<K:v\in\mathcal{I}_k} f(x_v^k) - f(x^\star)\,,\end{aligned}$$

where $\tau_{\max}$ is an upper bound on the maximal compute delay defined as $\tau_{\max} \ge \sup_{k<K}\sum_{\ell=k}^{\mathrm{next}(v,k+1)-1}|\mathcal{I}_\ell|$.

Thus,

$$\begin{aligned}\frac{2\gamma}{n}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k) - f(x^\star)) &\le -\mathbb{E}\left[\left\|\hat{x}^{k+1} - x^\star\right\|^2\right] + \mathbb{E}\left[\left\|\hat{x}^k - x^\star\right\|^2\right]\\ &+ \frac{\gamma^2\sigma^2|\mathcal{I}_k|}{n^2}\\ &+ \frac{2\gamma^2L|\mathcal{I}_k|}{n^2}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k) - f(x^\star))\\ &+ \frac{2LC\gamma}{n}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[(f(x_v^k) - f(x^\star))\right]\\ &+ \frac{\gamma}{Cn}\mathbb{E}\left[\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2\right]\\ &+ \frac{2\gamma^2L}{Cn^2}|\mathcal{I}_k|\sigma^2 + \frac{\gamma}{n}\Big(2LC\sum_{v\in\mathcal{I}_k}(f(x_v^k) - f(x^\star)) + \frac{2\gamma^2}{Cn}|\mathcal{I}_k|\sum_{v\in\mathcal{V}}\left\|\nabla f(x_v^{k-\tau(v,k)})\right\|^2\Big)\,.\end{aligned}$$

Summing over $k < K$, using Lemma 5.B.2 and our bound on T_3, we obtain:

$$\begin{aligned}
\frac{2\gamma}{n}\sum_{k<K}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k)-f(x^\star)) &\leq \mathbb{E}\left[\left\|\hat{x}^0-x^\star\right\|^2\right]+\frac{3\gamma^2\sigma^2}{n^2}\sum_{k<K}|\mathcal{I}_k| \\
&+\frac{2\gamma L}{n}\left(2C+\gamma+\frac{2\tau_{\max}\gamma^2}{Cn}\right)\sum_{k<K}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k)-f(x^\star)) \\
&+\sum_{k<K}\frac{\gamma}{Cn}\mathbb{E}\left[\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\right] \\
&\leq \mathbb{E}\left[\left\|\hat{x}^0-x^\star\right\|^2\right]+\frac{\gamma^2\sigma^2}{n^2}\left(1+\frac{2\gamma\bar{\rho}^{-1}n}{C}\right)\sum_{k<K}|\mathcal{I}_k|+\sum_{k<K}\frac{\gamma}{Cn}\mathbb{E}\left[\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\right] \\
&+\frac{\gamma^3B^2}{Cn}\sum_{k<K}|\mathcal{I}_k| \\
&+\frac{2\gamma L}{n}\left(2C+\gamma+\frac{2\tau_{\max}\gamma^2}{Cn}+\frac{4\gamma^2}{\bar{\rho}^2}\right)\sum_{k<K}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k)-f(x^\star))\,,
\end{aligned}$$

using Lemma 5.E.1 to handle the sum of the terms $\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2$.

Hence, provided that $2C+\gamma+\frac{2\tau_{\max}\gamma^2}{Cn}+\frac{4\gamma^2}{\bar{\rho}^2}\leq\frac{1}{2L}$, which is verified for $C=\frac{1}{8L}$ and $\gamma\leq\frac{1}{4L}\times\frac{1}{1+2\bar{\rho}^{-1}+4\sqrt{\tau_{\max}/n}}$, we have:

$$\frac{\gamma}{n}\sum_{k<K}\sum_{v\in\mathcal{I}_k}(\mathbb{E}f(x_v^k)-f(x^\star))\leq\mathbb{E}\left[\left\|\hat{x}^0-x^\star\right\|^2\right]+\frac{\gamma^2\sigma^2}{n^2}\left(3+16L\gamma\bar{\rho}^{-1}n\right)\sum_{k<K}|\mathcal{I}_k|\,,$$

leading to, for $\eta=\gamma/n$:

$$\mathbb{E}\left[f\left(\frac{1}{\sum_{k<K}|\mathcal{I}_k|}\sum_{k=0}^{K-1}\sum_{v\in\mathcal{I}_k}x_v^k\right)-f(x^\star)\right]\leq\frac{\mathbb{E}\left[\left\|\hat{x}^0-x^\star\right\|^2\right]}{\eta\sum_{k<K}|\mathcal{I}_k|}+3\eta\sigma^2+\eta^2 16L\sigma^2n^2\bar{\rho}^{-1}\,.$$

Optimizing over $\eta\leq\frac{1}{4L}\times\frac{1}{n(1+2\bar{\rho}^{-1})+4\sqrt{n\tau_{\max}}}$, we thus obtain that:

$$\begin{aligned}
&\mathbb{E}\left[f\left(\frac{1}{\sum_{k<K}|\mathcal{I}_k|}\sum_{k=0}^{K-1}\sum_{v\in\mathcal{I}_k}x_v^k\right)-f(x^\star)\right] \\
=&\mathcal{O}\left(\frac{LD^2(n\bar{\rho}^{-1}+\sqrt{n\tau_{\max}})}{\sum_{k<K}|\mathcal{I}_k|}+\sqrt{\frac{D\sigma^2}{\sum_{k<K}|\mathcal{I}_k|}}+\left[\frac{D^2\sqrt{L\sigma^2n^2\bar{\rho}^{-1}}}{\sum_{k<K}|\mathcal{I}_k|}\right]^{2/3}\right)\,.
\end{aligned}$$

□

5.F.2. Heterogeneous setting under sampling

Proof. As in the Lipschitz case, we have:

$$\begin{aligned}
\frac{2\gamma n\bar{p}}{n}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k)-f(x^\star)\right)&\leq 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0-\mathbf{x}^\star\right\|^2\right]+\frac{2\gamma^2(\sigma^2+2\zeta^2)\bar{p}K}{n}+4\gamma^2\bar{p}\sum_{k<K}\mathbb{E}\left[\left\|\nabla f(\bar{x}^k)\right\|^2\right] \\
&+\mathbb{E}\left[\frac{6\gamma Lp_{\max}}{n}\sum_{k<K}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\right]+8\gamma L\bar{p}\mathbb{E}\left[\sum_{k<K}\left\|\bar{x}^k-\hat{x}^k\right\|^2\right]\,.
\end{aligned}$$

Since losses are no longer assumed to be Lipschitz, we cannot bound this last term $\mathbb{E}\left[\sum_{k<K}\left\|\bar{x}^k-\hat{x}^k\right\|^2\right]$ by γ^2B^2. However, using Lemma 5.B.1,

$$\mathbb{E}\left[\sum_{k<K}\left\|\bar{x}^k-\hat{x}^k\right\|^2\right]\leq\frac{2\gamma^2\sigma^2K}{n}+\frac{2\gamma^2}{n}\mathbb{E}\left[\sum_{v\in\mathcal{V}}\sum_{k<K}\left\|\nabla f_v(x_v^{\mathrm{prev}(v,k)})\right\|^2\right].$$

Then,

$$\begin{aligned}\mathbb{E}\left[\sum_{v\in\mathcal{V}}\sum_{k<K}\left\|\nabla f_v(x_v^{\mathrm{prev}(v,k)})\right\|^2\right]&=\sum_{v\in\mathcal{V}}\sum_{k<K}\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\mathbb{1}_{v\in\mathcal{I}_k}(\mathrm{next}(k,v)-k)\right]\\&=\sum_{v\in\mathcal{V}}\sum_{k<K}\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\times\frac{1}{p_v}\times p_v\right]\\&=\sum_{v\in\mathcal{V}}\sum_{k<K}\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\right]\\&\leq\frac{1}{p_{\min}}\sum_{v\in\mathcal{V}}\sum_{k<K}p_v\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\right].\end{aligned}$$

since the random variables $\left\|\nabla f_v(x_v^k)\right\|^2$, $\mathbb{1}_{v\in\mathcal{I}_k}$ and $\mathrm{next}(k,v)-k$ are independent, $\mathbb{E}\left[\mathbb{1}_{v\in\mathcal{I}_k}\right]=p_v$ (Bernoulli random variable) and $\mathbb{E}\left[\mathrm{next}(k,v)-k\right]=\frac{1}{p_v}$ (geometric random variable). And then, as we proved before, $\sum_{v\in\mathcal{V}}\sum_{k<K}p_v\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\right]\leq 2L^2p_{max}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+2n\bar{p}\left\|\nabla f(\bar{x}^k)\right\|^2+2n\bar{p}\zeta^2$. Consequently,

$$\begin{aligned}\frac{2\gamma n\bar{p}}{n}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k)-f(x^\star)\right)&\leq 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0-\mathbf{x}^\star\right\|^2\right]+\frac{2\gamma^2(\sigma^2+2\zeta^2)\bar{p}K}{n}\\&\quad+(4\gamma^2\bar{p}+32\gamma^3L\bar{p}\frac{p_{\max}}{p_{\min}})\sum_{k<K}\mathbb{E}\left[\left\|\nabla f(\bar{x}^k)\right\|^2\right]\\&\quad+\mathbb{E}\left[(\frac{6\gamma Lp_{\max}}{n}+\frac{32\gamma^3L^3\bar{p}p_{\max}}{np_{\min}})\sum_{k<K}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\right]+\frac{16\gamma^3\sigma^2L\bar{p}K}{n}+\frac{32\gamma^3L\zeta^2\bar{p}^2}{p_{\min}}\\&\leq 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0-\mathbf{x}^\star\right\|^2\right]+\frac{2\gamma^2(\sigma^2+2\zeta^2)\bar{p}K}{n}\\&\quad+(4\gamma^2\bar{p}+32\gamma^3L\bar{p}\frac{p_{\max}}{p_{\min}})\sum_{k<K}\mathbb{E}\left[\left\|\nabla f(\bar{x}^k)\right\|^2\right]\\&\quad+\mathbb{E}\left[\frac{12\gamma Lp_{\max}}{n}\sum_{k<K}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\right]+\frac{16\gamma^3\sigma^2L\bar{p}K}{n}+\frac{32\gamma^3L\zeta^2\bar{p}^2}{p_{\min}}.\end{aligned}$$

provided that $\gamma\leq\sqrt{\frac{6p_{\min}}{32L^2p_{\max}}}$. Plugging Lemma 5.E.1 in here, we obtain:

$$\begin{aligned}\frac{2\gamma n\bar{p}}{n}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k)-f(x^\star)\right)&\leq 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0-\mathbf{x}^\star\right\|^2\right]+\frac{2\gamma^2(\sigma^2+2\zeta^2)\bar{p}K}{n}\\&\quad+(4\gamma^2\bar{p}+32\gamma^3L\bar{p}\frac{p_{\max}}{p_{\min}})\sum_{k<K}\mathbb{E}\left[\left\|\nabla f(\bar{x}^k)\right\|^2\right]\\&\quad+\frac{16\gamma^3\sigma^2L\bar{p}K}{n}+\frac{32\gamma^3L\zeta^2\bar{p}^2}{p_{\min}}\end{aligned}$$

$$
\begin{aligned}
&+ \frac{12\gamma L p_{\max}}{n}\left[4\gamma^2\sigma^2\bar{\rho}^{-1}n\bar{p}K + 8\gamma^2\bar{\rho}^{-2}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2 + 16\gamma^2\bar{\rho}^{-2}n\bar{p}\sum_{k<K}\left(\left\|\nabla f(\bar{x}^k)\right\|^2 + \zeta^2\right)\right]\\
&= 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0 - \mathbf{x}^\star\right\|^2\right] + \left(8\gamma^2 L\bar{p} + 192\gamma^3L^2p_{\max}\bar{p}\bar{\rho}^{-2} + 64\gamma^3L^2\bar{p}\frac{p_{\max}}{p_{\min}}\right)\sum_{k<K}\mathbb{E}\left[f(\bar{x}^k) - f(x^\star)\right]\\
&+ \frac{2\gamma^2(\sigma^2+2\zeta^2)\bar{p}K}{n} + \gamma^3K\left(8LB^2\bar{p} + 24L\sigma^2p_{\max}\bar{p}\bar{\rho}^{-1} + 96L\zeta^2p_{\max}\bar{p}\bar{\rho}^{-2}\right)\\
&+ \frac{96\gamma^3Lp_{\max}\bar{\rho}^{-2}}{n}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2.
\end{aligned}
$$

For $8\gamma^2L\bar{p}+192\gamma^3L^2p_{\max}\bar{p}\bar{\rho}^{-2}+64\gamma^3L^2\bar{p}\frac{p_{\max}}{p_{\min}} \leq \gamma\bar{p}$ which is verified for $\gamma \leq \min\left(\frac{1}{24L}, \frac{\bar{\rho}}{24L\sqrt{p_{\max}}}, \frac{1}{14L\sqrt{\frac{p_{\max}}{p_{\min}}}}\right)$, we have:

$$
\begin{aligned}
\gamma\bar{p}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k) - f(x^\star)\right) &\leq 2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0 - \mathbf{x}^\star\right\|^2\right] + \frac{96\gamma^3Lp_{\max}\bar{\rho}^{-2}}{n}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2\\
&\quad + \frac{2\gamma^2(\sigma^2+2\zeta^2)\bar{p}K}{n} + \gamma^3K\left(8LB^2\bar{p} + 24L\sigma^2p_{\max}\bar{p}\bar{\rho}^{-1} + 96L\zeta^2p_{\max}\bar{p}\bar{\rho}^{-2}\right),
\end{aligned}
$$

and thus:

$$
\begin{aligned}
\frac{1}{K}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k) - f(x^\star)\right) &\leq \frac{2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0 - \mathbf{x}^\star\right\|^2\right]}{\bar{p}\gamma K} + \frac{96\gamma^2Lp_{\max}\bar{\rho}^{-2}}{n\bar{p}K}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2\\
&\quad + \frac{2\gamma(\sigma^2+2\zeta^2)}{n} + \gamma^2\left(8LB^2 + 24L\sigma^2p_{\max}\bar{\rho}^{-1} + 96L\zeta^2p_{\max}\bar{\rho}^{-2}\right).
\end{aligned}
$$

Now, we use

$$
\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2 \leq \sum_{v\in\mathcal{V}}\left\|\nabla f(\bar{x}^0)\right\|^2 + \zeta^2 \leq \sum_{v\in\mathcal{V}}2L(f(x_0) - f(x^\star)) + \zeta^2,
$$

so that

$$
\begin{aligned}
&\frac{96\gamma^2Lp_{\max}\bar{\rho}^{-2}}{n\bar{p}K}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2 \leq \frac{192\gamma^2L^2p_{\max}\bar{\rho}^{-2}}{\bar{p}K}(f(x_0) - f(x^\star)) + \frac{96\zeta^2\gamma^2Lp_{\max}\bar{\rho}^{-2}}{\bar{p}K}\\
&\leq \frac{192\gamma^2L^2p_{\max}\bar{\rho}^{-2}}{\bar{p}}\frac{1}{K}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k) - f(x^\star)\right) + \frac{96\zeta^2\gamma^2Lp_{\max}\bar{\rho}^{-2}}{\bar{p}K}\\
&\leq \frac{1}{2}\frac{1}{K}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k) - f(x^\star)\right) + 96\zeta^2\gamma^2Lp_{\max}\bar{\rho}^{-2},
\end{aligned}
$$

for $K \geq \frac{1}{\bar{p}}$ and $\gamma \leq \frac{1\bar{\rho}}{384L}\sqrt{\frac{\bar{p}}{p_{\max}}}$. Thus,

$$
\begin{aligned}
\frac{1}{2K}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k) - f(x^\star)\right) &\leq \frac{2\mathbb{E}\left[\left\|\hat{\mathbf{x}}^0 - \mathbf{x}^\star\right\|^2\right]}{\bar{p}\gamma K}\\
&\quad + \frac{2\gamma(\sigma^2+2\zeta^2)}{n} + \gamma^2\left(8LB^2 + 24L\sigma^2p_{\max}\bar{\rho}^{-1} + 192L\zeta^2p_{\max}\bar{\rho}^{-2}\right).
\end{aligned}
$$

Optimizing over admissible γ's leads to:

$$\frac{1}{K}\sum_{k<K}\left(\mathbb{E}f(\bar{x}^k)-f(x^\star)\right)=\mathcal{O}\left(\frac{LD^2\left(\frac{1}{\bar{p}}\sqrt{\frac{p_{\max}}{p_{\min}}}+\sqrt{\frac{p_{\max}}{\bar{p}^2}}\bar{\rho}^{-1}\right)}{K}\right.$$
$$\left.+\sqrt{\frac{D^2(\sigma^2+\zeta^2)}{n\bar{p}K}}+\left[\frac{D^2\sqrt{LB^2+L\sigma^2p_{\max}\bar{\rho}^{-1}+L\zeta p_{\max}\bar{\rho}^{-2}}}{\bar{p}K}\right]^{\frac{2}{3}}\right).$$

□

5.G. Proof of Theorem 5.4: smooth non-convex case

5.G.1. Homogeneous without sampling

Proof. Using L-smoothness and a virtual sequence $\hat{x}$ defined in Section 5.B.1, we have

$$\mathbb{E}_{k+1}f(\hat{x}^{k+1})\leq f(\hat{x}^k)-\underbrace{\frac{\gamma}{n}\sum_{v\in\mathcal{I}_k}\left\langle\nabla f(\hat{x}^k),\nabla f(x_v^k)\right\rangle}_{:=T_1} \tag{5.69}$$
$$+\frac{L\gamma^2}{2n^2}\left(\sigma^2|\mathcal{I}_k|+\mathbb{E}\left\|\sum_{v\in\mathcal{I}_k}\nabla f(x_v^k)\right\|^2\right) \tag{5.70}$$

We separately estimate the middle term as

$$\begin{aligned}
T_1&=-\frac{\gamma}{n}\sum_{v\in\mathcal{I}_k}\left\langle\nabla f(\hat{x}^k),\nabla f(x_v^k)\right\rangle\\
&=-\frac{\gamma}{n}\sum_{v\in\mathcal{I}_k}\left\langle\nabla f(\bar{x}^k),\nabla f(x_v^k)\right\rangle+\frac{\gamma}{n}\sum_{v\in\mathcal{I}_k}\left\langle\nabla f(\bar{x}^k)-\nabla f(\hat{x}^k),\nabla f(x_v^k)\right\rangle\\
&\leq\frac{\gamma}{n}\sum_{v\in\mathcal{I}_k}\left(-\frac{1}{2}\left\|\nabla f(\bar{x}^k)\right\|^2-\frac{1}{2}\left\|\nabla f(x_v^k)\right\|^2+\frac{L^2}{2}\left\|x_v^k-\bar{x}^k\right\|^2\right)\\
&\quad+\frac{\gamma}{n}\sum_{v\in\mathcal{I}_k}\left(\frac{1}{4}\left\|\nabla f(x_v^k)\right\|^2+L^2\left\|\bar{x}^k-\hat{x}^k\right\|^2\right)\\
&\leq-\frac{\gamma}{4n}\sum_{v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2-\frac{|\mathcal{I}_k|\gamma}{2n}\left\|\nabla f(\bar{x}^k)\right\|^2\\
&\quad+\frac{L^2\gamma}{2n}\sum_{v\in\mathcal{I}_k}\left\|x_v^k-\bar{x}^k\right\|^2+\frac{\gamma L^2|\mathcal{I}_k|}{n}\left\|\bar{x}^k-\hat{x}^k\right\|^2
\end{aligned}$$

where we used that for any vectors $a,b\in\mathbb{R}^d$ it holds that $-\langle a,b\rangle=-\frac{1}{2}\|a\|^2-\frac{1}{2}\|b\|^2+\frac{1}{2}\|a-b\|^2$ and also it holds that $2\langle a,b\rangle\leq\gamma\|a\|^2+\gamma^{-1}\|b\|^2$ for any $\gamma>0$ and we chose $\gamma=2$.

We further use Lemma 5.B.1 to estimate the last term

$$T_1\leq-\frac{\gamma}{4n}\sum_{v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2-\frac{|\mathcal{I}_k|\gamma}{2n}\left\|\nabla f(\bar{x}^k)\right\|^2+\frac{L^2\gamma}{2n}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2$$
$$+\frac{2L^2\gamma^3|\mathcal{I}_k|}{n^2}\left(\sigma^2+\sum_{v\in\mathcal{V}}\mathbb{E}\left[\left\|\nabla f_v(x_v^{\mathrm{prev}(v,k)})\right\|^2\right]\right)$$

Putting this estimate of T_1 back into (5.69) we get

$$\begin{aligned}\mathbb{E}_{k+1} f(\hat{x}^{k+1}) \leq f(\hat{x}^k) + \frac{L\gamma^2\sigma^2|\mathcal{I}_k|}{2n^2} + \frac{L\gamma^2}{2n}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left\|\nabla f(x_v^k)\right\|^2 \\ - \frac{\gamma}{4n}\sum_{v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2 - \frac{|\mathcal{I}_k|\gamma}{2n}\left\|\nabla f(\bar{x}^k)\right\|^2 \\ + \frac{L^2\gamma}{2n}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 + \frac{2L^2\gamma^3|\mathcal{I}_k|}{n^2}\left(\sigma^2 + \sum_{v\in\mathcal{V}}\mathbb{E}\left[\left\|\nabla f_v(x_v^{\mathrm{prev}(v,k)})\right\|^2\right]\right)\end{aligned}$$

Using that $\gamma < \frac{1}{4L}$ we estimate

$$\begin{aligned}\mathbb{E}_{k+1} f(\hat{x}^{k+1}) \leq f(\hat{x}^k) - \frac{\gamma}{8n}\sum_{v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2 - \frac{|\mathcal{I}_k|\gamma}{2n}\left\|\nabla f(\bar{x}^k)\right\|^2 + \frac{L^2\gamma}{2n}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 \\ + \frac{2L^2\gamma^3|\mathcal{I}_k|}{n^2}\left(\sigma^2 + \sum_{v\in\mathcal{V}}\mathbb{E}\left[\left\|\nabla f_v(x_v^{\mathrm{prev}(v,k)})\right\|^2\right]\right) + \frac{L\gamma^2\sigma^2|\mathcal{I}_k|}{2n^2}\end{aligned}$$

Taking the full expectation and summing over all the iterations k, we get

$$\begin{aligned}\sum_{k<K}\frac{|\mathcal{I}_k|\gamma}{2n}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 \leq (f(x^0) - f^\star) - \frac{\gamma}{8n}\sum_{k<K}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left\|\nabla f(x_v^k)\right\|^2 + \frac{L^2\gamma}{2n}\sum_{k<K}\mathbb{E}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2 \\ + \frac{L\gamma^2\sigma^2\sum_{k<K}|\mathcal{I}_k|}{2n^2}(1+4L\gamma) + \frac{2L^2\gamma^3}{n^2}\sum_{k<K}\sum_{v\in\mathcal{V}}|\mathcal{I}_k|\mathbb{E}\left[\left\|\nabla f_v(x_v^{\mathrm{prev}(v,k)})\right\|^2\right]\end{aligned}$$

For the third term we use Lemma 5.B.2, and for the last term we use that

$$\begin{aligned}\sum_{k<K}|\mathcal{I}_k|\sum_{v\in\mathcal{V}}\left\|\nabla f(x_v^{\mathrm{prev}(v,k)})\right\|^2 &\leq \sum_{v\in\mathcal{V}}\sum_{k<K:v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2\sum_{\ell=k}^{\mathrm{next}(v,k+1)-1}|\mathcal{I}_\ell| \\ &\leq \tau_{\max}\sum_{v\in\mathcal{V}}\sum_{k<K:v\in\mathcal{I}_k}\left\|\nabla f(x_v^k)\right\|^2\end{aligned}$$

where $\tau_{\max}$ is an upper bound on the maximal compute delay defined as $\tau_{\max} \geq \sup_{k<K}\sum_{\ell=k}^{\mathrm{next}(v,k+1)-1}|\mathcal{I}_\ell|$. For estimating the third term with Lemma 2, we also use that

$$\sum_{k<K}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[\left\|\nabla f_v(x_v^{k-\tau(k,v)})\right\|^2\right] \leq \sum_{k<K}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\right]$$

We therefore get

$$\begin{aligned}\sum_{k<K}\frac{|\mathcal{I}_k|\gamma}{2n}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 \leq (f(x^0) - f^\star) - \frac{\gamma}{8n}\sum_{k<K}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left\|\nabla f(x_v^k)\right\|^2 + \frac{L^2\gamma}{2n}2\gamma^2\sigma^2\bar{\rho}^{-1}\sum_{k<K}|\mathcal{I}_k| \\ + \frac{2L^2\gamma^3}{n\bar{\rho}^2}\sum_{k<K}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left[\left\|\nabla f(x_v^k)\right\|^2\right] \\ + \frac{L\gamma^2\sigma^2\sum_{k<K}|\mathcal{I}_k|}{2n^2}(1+4L\gamma) + \frac{2L^2\gamma^3}{n^2}\tau_{\max}\sum_{k<K}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left\|\nabla f(x_v^k)\right\|^2\end{aligned}$$

We further use that the stepsize $\gamma < \frac{1}{8L}(\sqrt{\frac{n}{\tau_{\max}}} + \bar{\rho})$

$$\sum_{k<K} \frac{|\mathcal{I}_k|\gamma}{2n}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 \leq (f(x^0) - f^\star) + \frac{L^2}{n}\gamma^3\sigma^2\bar{\rho}^{-1}\sum_{k<K}|\mathcal{I}_k| + \frac{L\gamma^2\sigma^2\sum_{k<K}|\mathcal{I}_k|}{n^2}$$

Therefore,

$$\sum_{k<K} |\mathcal{I}_k|\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 \leq \frac{2n}{\gamma}(f(x^0) - f^\star) + 2L^2\gamma^2\sigma^2\bar{\rho}^{-1}\sum_{k<K}|\mathcal{I}_k| + \frac{2L\gamma\sigma^2\sum_{k<K}|\mathcal{I}_k|}{n}$$

Denoting $T = \sum_{k<K}|\mathcal{I}_k|$ and tuning over the stepsize γ, we get

$$\frac{1}{\sum_{k<K}|\mathcal{I}_k|}\sum_{k<K}|\mathcal{I}_k|\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 \leq \frac{16LF_0(\sqrt{n\tau_{\max}} + n\bar{\rho}^{-1})}{T} + 4\left(\frac{L\sigma^2F_0}{T}\right)^{\frac{1}{2}} + 4\left(\frac{L\sigma nF_0}{T\sqrt{\bar{\rho}}}\right)^{\frac{2}{3}}$$

where $F_0 = (f(x^0) - f^\star)$. □

5.G.2. Heterogeneous with sampling

Proof. Using L-smoothness of f,

$$\mathbb{E}_{k+1}f(\hat{x}^{k+1}) \leq f(\hat{x}^k) - \underbrace{\frac{\gamma}{n}\mathbb{E}\sum_{v\in\mathcal{I}_k}\left\langle\nabla f(\hat{x}^k), \nabla f_v(x_v^k)\right\rangle}_{:=T_1} \tag{5.71}$$

$$+ \frac{L\gamma^2}{2n^2}\left(\sigma^2\mathbb{E}|\mathcal{I}_k| + \mathbb{E}\left\|\sum_{v\in\mathcal{I}_k}\nabla f_v(x_v^k)\right\|^2\right) \tag{5.72}$$

We separately estimate the T_1 term

$$\begin{aligned}
T_1 &= -\frac{\gamma}{n}\mathbb{E}\sum_{v\in\mathcal{I}_k}\left\langle\nabla f(\hat{x}^k), \nabla f_v(x_v^k)\right\rangle \\
&= -\frac{\gamma}{n}\mathbb{E}\sum_{v\in\mathcal{I}_k}\left\langle\nabla f(\hat{x}^k), \nabla f_v(\bar{x}^k)\right\rangle + \frac{\gamma}{n}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left\langle\nabla f(\hat{x}^k), \nabla f_v(\bar{x}^k) - \nabla f_v(x_v^k)\right\rangle \\
&= -\gamma\bar{p}\left\langle\nabla f(\hat{x}^k), \nabla f(\bar{x}^k)\right\rangle + \frac{\gamma}{n}\sum_{v\in\mathcal{I}_k}\mathbb{E}\left\langle\nabla f(\hat{x}^k), \nabla f_v(\bar{x}^k) - \nabla f_v(x_v^k)\right\rangle \\
&\leq \gamma\bar{p}\left(-\frac{1}{2}\left\|\nabla f(\hat{x}^k)\right\|^2 - \frac{1}{2}\left\|\nabla f(\bar{x}^k)\right\|^2 + \frac{L^2}{2}\left\|\hat{x}^k - \bar{x}^k\right\|^2\right) \\
&\quad + \frac{\gamma}{n}\left(\frac{1}{2}n\bar{p}\left\|\nabla f(\hat{x}^k)\right\|^2 + \frac{L^2}{2}\mathbb{E}\sum_{v\in\mathcal{I}_k}\left\|x_v^k - \bar{x}^k\right\|^2\right)
\end{aligned}$$

Since $\mathbb{E}\sum_{v\in\mathcal{I}_k}\nabla f_v(\bar{x}^k) = n\bar{p}\,\nabla f(\bar{x}^k)$, and $\mathbb{E}|\mathcal{I}_k| = n\bar{p}$. We further use that $\mathbb{E}\sum_{v\in\mathcal{I}_k}\left\|x_v^k - \bar{x}^k\right\|^2 = \sum_{v\in\mathcal{V}}p_v\left\|x_v^k - \bar{x}^k\right\|^2 \leq p_{\max}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2$. Therefore,

$$T_1 \leq -\frac{\gamma\bar{p}}{2}\left\|\nabla f(\bar{x}^k)\right\| + \frac{\gamma\bar{p}L^2}{2}\left\|\hat{x}^k - \bar{x}^k\right\|^2 + \frac{\gamma L^2 p_{\max}}{2n}\left\|\mathbf{x}^k - \bar{\mathbf{x}}^k\right\|^2$$

Putting this back to (5.71) and summing it up over K, we get

$$\frac{\gamma\bar{p}}{2}\sum_{k<K}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 \leq (f(x^0)-f^\star)+\frac{\gamma\bar{p}L^2}{2}\sum_{k<K}\mathbb{E}\left\|\hat{x}^k-\bar{x}^k\right\|^2 + \frac{\gamma L^2 p_{\max}}{2n}\sum_{k<K}\mathbb{E}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+\frac{L\gamma^2\sigma^2\bar{p}K}{2n} + \frac{L\gamma^2}{2n^2}\sum_{k<K}\mathbb{E}\left\|\sum_{v\in\mathcal{I}_k}\nabla f_v(x_v^k)\right\|^2$$

We use calculations from Section 5.E.2 to further estimate the last term

$$\mathbb{E}\left\|\sum_{v\in\mathcal{I}_k}\nabla f_v(x_v^k)\right\|^2 \leq \sum_{v\in\mathcal{V}}p_v\left\|\nabla f_v(x_v^k)\right\|^2+\left\|\sum_{v\in\mathcal{V}}p_v\nabla f_v(x_v^k)\right\|^2$$

$$\sum_{v\in\mathcal{V}}p_v\left\|\nabla f_v(x_v^k)\right\|^2 \leq 2L^2p_{max}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+2n\bar{p}\left\|\nabla f(\bar{x}^k)\right\|^2+2n\bar{p}\zeta^2\,. \tag{5.73}$$

$$\left\|\sum_{v\in\mathcal{V}}p_v\nabla f_v(x_v^k)\right\|^2 \leq 2(n\bar{p})^2\left\|\nabla f(\bar{x}^k)\right\|^2+2(n\bar{p})p_{\max}L^2\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2$$

We therefore get

$$\begin{aligned}
&\frac{\gamma\bar{p}}{2}\sum_{k<K}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 \leq (f(x^0)-f^\star)+\frac{\gamma\bar{p}L^2}{2}\sum_{k<K}\mathbb{E}\left\|\hat{x}^k-\bar{x}^k\right\|^2\\
&\quad+\frac{\gamma L^2p_{\max}}{2n}\sum_{k<K}\mathbb{E}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+\frac{L\gamma^2\sigma^2\bar{p}K}{2n}\\
&\quad+\frac{L\gamma^2}{n^2}\left((L^2p_{\max}+(n\bar{p})p_{\max}L^2)\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+(n\bar{p}+(n\bar{p})^2)\left\|\nabla f(\bar{x}^k)\right\|^2+n\bar{p}\zeta^2\right)\\
&\leq (f(x^0)-f^\star)+\frac{\gamma\bar{p}L^2}{2}\sum_{k<K}\mathbb{E}\left\|\hat{x}^k-\bar{x}^k\right\|^2\\
&\quad+\frac{\gamma L^2p_{\max}}{2n}\left[1+2L\gamma\left(\frac{1}{n}+\bar{p}\right)\right]\sum_{k<K}\mathbb{E}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\\
&\quad+\frac{L\gamma^2\bar{p}K\sigma^2}{2n}+\frac{L\gamma^2n\bar{p}(1+n\bar{p})}{n^2}\sum_{k<K}\left\|\nabla f(\bar{x}^k)\right\|^2+\frac{L\gamma^2\bar{p}\zeta^2K}{n}
\end{aligned}$$

We further use Lemma 5.B.1 to estimate the term with $\mathbb{E}\left\|\hat{x}^k-\bar{x}^k\right\|^2$:

$$\mathbb{E}\left[\left\|\hat{x}^k-\bar{x}^k\right\|^2\right] \leq \frac{2\gamma^2}{n}\left(\sigma^2+\sum_{v\in\mathcal{V}}\mathbb{E}\left[\left\|\nabla f_v(x_v^{\mathrm{prev}(v,k)})\right\|^2\right]\right)$$

And we use calculations from Section 5.F.2 estimating

$$\mathbb{E}\left[\sum_{v\in\mathcal{V}}\sum_{k<K}\left\|\nabla f_v(x_v^{\text{prev}(v,k)})\right\|^2\right] \leq \frac{1}{p_{\min}}\sum_{v\in\mathcal{V}}\sum_{k<K} p_v\mathbb{E}\left[\left\|\nabla f_v(x_v^k)\right\|^2\right].$$

And (5.73) to estimate the last term. Therefore we get

$$\begin{aligned}&\sum_{k<K}\mathbb{E}\left[\left\|\hat{x}^k-\bar{x}^k\right\|^2\right]\\ &\leq \frac{2\gamma^2}{n}\left(\sigma^2K+\frac{1}{p_{\min}}\sum_{k<K}2L^2p_{max}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2+2n\bar{p}\sum_{k<K}\left\|\nabla f(\bar{x}^k)\right\|^2+2n\bar{p}\zeta^2K\right)\end{aligned}$$

And

$$\begin{aligned}\frac{\gamma\bar{p}}{2}\sum_{k<K}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 &\leq (f(x^0)-f^\star)\\ &+\frac{\gamma L^2p_{\max}}{2n}\left[1+2L\gamma\left(\frac{1}{n}+\bar{p}\right)+\frac{4\gamma^2L^2\bar{p}}{p_{\min}}\right]\sum_{k<K}\mathbb{E}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\\ &+\frac{L\gamma^2\bar{p}K\sigma^2}{2n}(1+\gamma L)+\frac{L\gamma^2n\bar{p}(1+n\bar{p}+2\gamma Ln\bar{p}/p_{\min})}{n^2}\sum_{k<K}\left\|\nabla f(\bar{x}^k)\right\|^2\\ &+\frac{L\gamma^2\bar{p}\zeta^2K}{n}(1+2n\bar{p}\gamma L)\end{aligned}$$

We further use that $\gamma < \min\left\{\frac{1}{4L}, \frac{\sqrt{\bar{p}}}{4L\sqrt{p_{\min}}}\right\}$

$$\begin{aligned}\frac{\gamma\bar{p}}{2}\sum_{k<K}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 &\leq (f(x^0)-f^\star)+\frac{\gamma L^2p_{\max}}{n}\sum_{k<K}\mathbb{E}\left\|\mathbf{x}^k-\bar{\mathbf{x}}^k\right\|^2\\ &+\frac{L\gamma^2\bar{p}K\sigma^2}{2n}(1+\gamma L)+3L\gamma^2\bar{p}\sum_{k<K}\left\|\nabla f(\bar{x}^k)\right\|^2\\ &+\frac{L\gamma^2\bar{p}\zeta^2K}{n}(1+2n\bar{p}\gamma L)\end{aligned}$$

We further use Lemma 5.E.1:

$$\begin{aligned}\frac{\gamma\bar{p}}{2}\sum_{k<K}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2 &\leq \frac{\gamma L^2p_{\max}}{n}\left[4\gamma^2\sigma^2\bar{\rho}^{-1}n\bar{p}K+8\gamma^2\bar{\rho}^{-2}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2\right.\\ &\left.+16\gamma^2\bar{\rho}^{-2}n\bar{p}\sum_{k<K}\left(\left\|\nabla f(\bar{x}^k)\right\|^2+\zeta^2\right)\right]\\ &+\frac{L\gamma^2\bar{p}K\sigma^2}{2n}(1+\gamma L)+3L\gamma^2\bar{p}\sum_{k<K}\left\|\nabla f(\bar{x}^k)\right\|^2\\ &+\frac{L\gamma^2\bar{p}\zeta^2K}{n}(1+2n\bar{p}\gamma L)+(f(x^0)-f^\star)\\ &\leq (f(x^0)-f^\star)+4\gamma^3L^2p_{\max}\bar{p}\bar{\rho}^{-1}K\sigma^2+\frac{8\gamma^3L^2p_{\max}\bar{\rho}^{-2}}{n}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2\end{aligned}$$

$$+\frac{L\gamma^2\bar{p}K\sigma^2}{2n}(1+\gamma L)$$
$$+L\gamma^2\bar{p}(3+16\gamma L\bar{\rho}^{-2}p_{\max})\sum_{k<K}\left\|\nabla f(\bar{x}^k)\right\|^2+\frac{2L\gamma^2\bar{p}\zeta^2K}{n}(1+8\rho^{-2}n\gamma Lp_{\max})$$

Taking the stepsize $\gamma<\min\{\frac{1}{24L},\frac{\bar{\rho}}{16L\sqrt{p_{\max}}}\}$ we get:

$$\frac{\gamma\bar{p}}{4}\sum_{k<K}\mathbb{E}\left\|\nabla f(\bar{x}^k)\right\|^2\leq(f(x^0)-f^\star)+4\gamma^3L^2p_{\max}\bar{p}\bar{\rho}^{-1}K\sigma^2$$
$$+\frac{8\gamma^3L^2p_{\max}\bar{\rho}^{-2}}{n}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2+\frac{2L\gamma^2\bar{p}K\sigma^2}{2n}$$
$$+\frac{2L\gamma^2\bar{p}\zeta^2K}{n}(1+8\rho^{-2}n\gamma Lp_{\max})$$

We conclude as in the smooth convex case by tuning the stepsize and getting rid of the $\frac{8\gamma^3L^2p_{\max}\bar{\rho}^{-2}}{n}\sum_{v\in\mathcal{V}}\left\|\nabla f_v(\bar{x}^0)\right\|^2$. □

Part II

Privacy and Personalization

Chapter 6

Muffliato: DIFFERENTIALLY-PRIVATE DECENTRALIZED LEARNING

In the first part of this manuscript, we have focused on asynchrony and decentralization, two components of optimization that are becoming increasingly popular in machine learning, for their scalability and efficiency.

Intuitively, decentralization should also provide better privacy guarantees, as nodes only observe the messages sent by their neighbors in the network graph. But formalizing and quantifying this gain is challenging: existing results are typically limited to Local Differential Privacy (LDP) guarantees that overlook the advantages of decentralization.

Informally, for a decentralized learning scheme over a graph $G = (\mathcal{V}, \mathcal{E})$, we would ideally like to exhibit privacy guarantees of the form *"node u is private with respect to node v with a privacy loss quantified by $\varepsilon_{u\to v}$"*, where the quantity $\varepsilon_{u\to v}$ quantifies privacy leaks from u to v, and decreases as the distance between nodes u and v in the graph increases.

In this chapter, we aim at filling this gap in the literature and introduce pairwise network differential privacy, a relaxation of LDP that captures the fact that the privacy leakage from a node u to a node v may depend on their relative position in the graph. We then analyze 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.

6.1. Introduction

Training machine learning models traditionally requires centralizing data in a single server, raising issues of scalability and privacy. An alternative is to use Federated Learning (FL), where each user keeps her data on device [MMR+17, KMA+19]. In *fully decentralized* FL, the common hypothesis of a central server is also removed, letting users, represented as nodes in a graph, train the model via peer-to-peer communications along edges. This approach improves scalability and robustness to central server failures, enabling lower latency, less power consumption and quicker deployment [LS07, BGPS06, SBB+17, NCTV19, AS19, LZZ+17, KLB+20].

Another important dimension is privacy, as a wide range of applications deal with sensitive and personal data. The gold standard to quantify the privacy leakage of algorithms is Differential Privacy (DP) [DR14]. DP typically requires to randomly perturb data-dependent computations to prevent the final model from leaking too much information about any in-

dividual data point (e.g., through data memorization). However, decentralized algorithms do not only reveal the final model to the participating nodes, but also the results of some intermediate computations. A standard solution is to use Local Differential Privacy (LDP) [KLN+08, DJW13], where random perturbations are performed locally by each user, thus protecting against an attacker that would observe everything that users share. From the algorithmic standpoint, local noise addition can be easily performed within decentralized algorithms, as done for instance in [HMV15, BGTT18, CYH+19, ZKL18, XZW21]. Unfortunately, LDP requires large amounts of noise, and thus provides poor utility.

In this chapter, we show that LDP guarantees give a very pessimistic view of the privacy offered by decentralized algorithms. Indeed, there is no central server receiving all messages, and the participating nodes can only observe the messages sent by their neighbors in the graph. So, a given node should intuitively leak less information about its private data to nodes that are far away. We formally quantify this 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.

Our results are 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.

6.1.1. Contributions and outline of the chapter

(i) Inspired by the recent definitions of [CB20], we introduce *pairwise network DP*, a relaxation of Local Differential Privacy which is able to quantify the privacy loss of a decentralized algorithm for each pair of distinct users in a graph.

(ii) 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.

(iii) We analyze both synchronous gossip [DKM+10] and randomized gossip [BGPS06] 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 λ_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.

(iv) 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

[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.

privacy amplification described above for averaging, up to factors that depend on the regularity of the global objective.

(v) 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.

6.1.2. Related work

Gossip algorithms and decentralized optimization. Gossip algorithms [BGPS06, DKM+10] were introduced to compute the global average of local vectors through peer-to-peer communication, and are at the core of many decentralized optimization algorithms. Classical decentralized optimization algorithms alternate between gossip communications and local gradient steps [NO09, KSJ19, KLB+20], or use dual formulations and formulate the consensus constraint using gossip matrices to obtain decentralized dual or primal-dual algorithms [SBB+17, HBM19, EHM20, EBB+21, KGGR21b, AS19]. We refer the reader to [NOR18] for a broader survey on decentralized optimization. Our algorithms are based on the general analysis of decentralized SGD in [KLB+20].

LDP and privacy amplification mechanisms. Limitations of LDP for computing the average of the private values of n users have been studied, showing that for a fixed privacy budget, the expected squared error in LDP is n times larger than in central DP [CSS12a]. More generally, LDP is also known to significantly reduce utility for many learning problems [ZMW17a, WGX18], which motivates the study of intermediate trust models. Cryptographic primitives, such as secure aggregation [DKM+06, SCR+11, BIK+17, CSS12b, JWEG18, BBG+20a, SBR20] and secure shuffling [CSU+19, EFM+19, BBGN19, GGK+20, FMT20], as well as additional mechanisms such as amplification by subsampling [BBG18] or amplification by iteration [FMTT18a], can offer better utility for some applications, but cannot be easily applied in a fully decentralized setting, as they require coordination by a central server.

Privacy amplification through decentralization. The idea that decentralized communications can provide differential privacy guarantees was initiated by [BGH20] in the context of rumor spreading. Closer to our work, [CB20] showed privacy amplification for random walk algorithms on complete graphs, where the model is transmitted from one node to another sequentially. While we build on their notion of Network DP, our work differs from [CB20] in several aspects: (i) our analysis holds for any graph and explicitly quantifies its effect, (ii) instead of worst-case privacy across all pairs of nodes, we prove pairwise guarantees that are stronger for nodes that are far away from each other, and (iii) unlike random walk approaches, gossip algorithms allow parallel computation and thus better scalability.

6.2. Setting and Pairwise Network Differential Privacy

We study a decentralized model where n nodes (users) hold private datasets and communicate through gossip protocols, that we describe in Section 6.2.1. In Section 6.2.2, we recall differential privacy notions and the two natural baselines for our work, central and local DP. Finally, we introduce in Section 6.2.3 the relaxation of local DP used throughout this chapter: the *pairwise network DP*.

6.2.1. Gossip Algorithms

We consider a connected graph $G = (\mathcal{V}, \mathcal{E})$ on a set $\mathcal{V}$ of n users. An edge $\{u, v\} \in \mathcal{E}$ indicates that u and v can communicate (we say they are neighbors). Each user $v \in \mathcal{V}$ holds a local dataset $\mathcal{D}_v$ and we aim at computing averages of private values. This averaging step

is a key building block for solving machine learning problems in a decentralized manner, as will be discussed in Section 6.4. From any graph, we can derive a gossip matrix.

Definition 6.2.1 (Gossip matrix). *A gossip matrix over a graph G is a* symmetric *matrix $W \in \mathbb{R}^{n\times n}$ with non-negative entries, that satisfies $W\mathbb{1} = \mathbb{1}$* i.e. *W is stochastic ($\mathbb{1} \in \mathbb{R}^n$ is the vector with all entries equal to 1), and such that for any $u, v \in \mathcal{V}$, $W_{u,v} > 0$ implies that $\{u, v\} \in \mathcal{E}$ or $u = v$.*

The iterates of synchronous gossip [DKM+10] are generated through a recursion of the form $x^{t+1} = Wx^t$, and converge to the mean of initial values $x^0 \in \mathbb{R}^n$ at a linear rate $e^{-t\lambda_W}$, with λ_W defined below.

Definition 6.2.2 (Spectral gap). *The spectral gap λ_W associated with a gossip matrix W is $\min_{\lambda\in \mathrm{Sp}(W)\setminus\{1\}}(1 - |\lambda|)$, where $\mathrm{Sp}(W)$ is the spectrum of W.*

The inverse of λ_W is the relaxation time of the random walk on G with transition probabilities W, and is closely related to the connectivity of the graph: adding edges improve mixing properties (λ_W increases), but can reduce scalability by increasing node degrees (and thus the per-iteration communication complexity). The rate of convergence can be accelerated to $e^{-t\sqrt{\lambda_W}}$ using re-scaled Chebyshev polynomials, leading to iterates of the form $x^t = P_t(W)x^0$ [BBG20b].

Definition 6.2.3 (Re-scaled Chebyshev polynomials). *The re-scaled Chebyshev polynomials $(P_t)_{t\geq 0}$ with scale parameter $\gamma \in [1, 2]$ are defined by second-order linear recursion:*

$$P_0(X) = 1\,, \quad P_1(X) = X\,, \quad P_{t+1}(X) = \gamma X P_t(X) + (1-\gamma)P_{t-1}(X)\,,\ t \geq 2\,. \tag{6.1}$$

6.2.2. Rényi Differential Privacy

Differential Privacy (DP) quantifies how much information the output of an algorithm $\mathcal{A}$ leaks about the dataset taken as input [DR14]. DP requires to define an adjacency relation between datasets. In this work, we adopt a user-level relation [MRTZ18] which aims to protect the whole dataset $\mathcal{D}_v$ of a given user represented by a node $v \in \mathcal{V}$. Formally, $\mathcal{D} = \cup_{v\in\mathcal{V}}\mathcal{D}_v$ and $\mathcal{D}' = \cup_{v\in\mathcal{V}}\mathcal{D}'_v$ are adjacent datasets, denoted by $\mathcal{D} \sim \mathcal{D}'$, if there exists $v \in \mathcal{V}$ such that only $\mathcal{D}_v$ and $\mathcal{D}'_v$ differ. We use $\mathcal{D} \sim_v \mathcal{D}'$ to denote that $\mathcal{D}$ and $\mathcal{D}'$ differ only in the data of user v.

We use Rényi Differential Privacy (RDP) [Mir17] to measure the privacy loss, which allows better and simpler composition than the classical (ϵ, δ)-DP. Note that any (α, ε)-RDP algorithm is also $(\varepsilon + \ln(1/\delta)/(\alpha - 1), \delta)$-DP for any $0 < \delta < 1$ [Mir17].

Definition 6.2.4 (Rényi Differential Privacy). *An algorithm $\mathcal{A}$ satisfies (α, ε)-Rényi Differential Privacy (RDP) for $\alpha > 1$ and $\varepsilon > 0$ if for all pairs of neighboring datasets $\mathcal{D} \sim \mathcal{D}'$:*

$$D_\alpha\left(\mathcal{A}(\mathcal{D})||\mathcal{A}(\mathcal{D}')\right) \leq \varepsilon\,, \tag{6.2}$$

where for two random variables X and Y, $D_\alpha\big(X\,||\,Y\big)$ is the Rényi divergence *between X and Y:*

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

with μ_X and μ_Y the respective densities of X and Y.

Without loss of generality, we consider gossip algorithms with a single real value per node (in that case, $\mathcal{D}_v = \{x_v\}$ for some $x_v \in \mathbb{R}$), and we aim at computing a private estimation of the mean $\bar{x} = (1/n)\sum_v x_v$. The generalization to vectors is straightforward, as done subsequently for optimization in Section 6.4. In general, the value of a (scalar) function g of the data can be privately released using the Gaussian mechanism [DR14, Mir17], which adds $\eta \sim \mathcal{N}(0, \sigma^2)$ to $g(\mathcal{D})$. It satisfies $(\alpha, \alpha\Delta_g^2/2\sigma^2)$-RDP for any $\alpha > 1$, where

$\Delta_g = \sup_{\mathcal{D}\sim\mathcal{D}'} \|g(\mathcal{D}) - g(\mathcal{D}')\|$ is the sensitivity of g. We focus on the Gaussian mechanism for its simplicity (similar results could be derived for other DP mechanisms), and thus assume an upper bound on the L_2 inputs sensitivity.

Assumption 6.2.1. *There exists some constant $\Delta > 0$ such that for all $u \in \mathcal{V}$ and for any adjacent datasets $\mathcal{D} \sim_u \mathcal{D}'$, we have $\|x_u - x'_u\| \leq \Delta$.*

The Gaussian mechanism then satisfies the following basic property.

Lemma 6.2.1 (Gaussian mechanism)**.** *For $\alpha > 1$, noise variance σ^2, sensitivity $\Delta > 0$ and $x, y \in \mathbb{R}$ such that $|x - y| \leq \Delta$, we have:*

$$D_\alpha\big(\mathcal{N}(x, \sigma^2) \,||\, \mathcal{N}(y, \sigma^2)\big) \leq \frac{\alpha\Delta^2}{2\sigma^2}\,.$$

In central DP, a trusted aggregator can first compute the mean $\bar{x}$ (which has sensitivity Δ/n) and then reveal a noisy version with the Gaussian mechanism. On the contrary, in local DP where there is no trusted aggregator and everything that a given node reveals can be observed, each node must locally perturb its input (which has sensitivity Δ), deteriorating the privacy-utility trade-off. Formally, to achieve (α, ε)-DP, one cannot have better utility than:

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

where x^{out} is the output of the algorithm. This $1/n$ gap motivates the study of relaxations of local DP.

6.2.3. Pairwise Network Differential Privacy

We relax local DP to take into account privacy amplification between nodes that are distant from each other in the graph. We define a decentralized algorithm $\mathcal{A}$ as a randomized mapping that takes as input a dataset $\mathcal{D} = \cup_{v\in\mathcal{V}}(\mathcal{D}_v)$ and outputs the transcript of all messages exchanged between users in the network. A message between neighboring users $\{u, v\} \in \mathcal{E}$ at time t is characterized by the tuple $(u, m(t), v)$: user u sent a message with content $m(t)$ to user v, and $\mathcal{A}(\mathcal{D})$ is the set of all these messages. Each node v only has a partial knowledge of $\mathcal{A}(\mathcal{D})$, captured by its *view*:

$$\mathcal{O}_v\big(\mathcal{A}(\mathcal{D})\big) = \{(u, m(t), v) \in \mathcal{A}(\mathcal{D}) \quad \text{such that} \quad \{u, v\} \in \mathcal{E}\}\,.$$

This subset corresponds to direct interactions of v with its neighbors, which provide only an indirect information on computations in others parts of the graph. Thus, we seek to express privacy constraints that are personalized for each pair of nodes. This is captured by our notion of Pairwise Network DP.

Definition 6.2.5 (Pairwise Network DP)**.** *For $f : \mathcal{V} \times \mathcal{V} \to \mathbb{R}^+$, an algorithm $\mathcal{A}$ satisfies (α, f)-Pairwise Network DP (PNDP) if for all pairs of distinct users $u, v \in \mathcal{V}$ and neighboring datasets $D \sim_u D'$:*

$$D_\alpha\big(\mathcal{O}_v(\mathcal{A}(\mathcal{D})) \,||\, \mathcal{O}_v(\mathcal{A}(\mathcal{D}'))\big) \leq f(u, v)\,. \tag{6.3}$$

We note $\varepsilon_{u\to v} = f(u, v)$ the privacy leaked to u from v and say that u is $(\alpha, \varepsilon_{u\to v})$-PNDP with respect to v if only inequality (6.3) holds for $f(u, v) = \varepsilon_{u\to v}$.

By taking f constant in Definition 6.2.5, we recover the definition of Network DP [CB20]. 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 assume that users are *honest but curious*: they truthfully follow the protocol, but may try to derive as much information

as possible from what they observe. We refer to Appendix 6.D for a natural adaptation of our definition and results to the presence of *colluding nodes* and to the protection of *groups* of users.

In addition to pairwise guarantees, we will use the *mean privacy loss*

$$\overline{\varepsilon}_v = \frac{1}{n} \sum_{u \in \mathcal{V} \setminus \{v\}} f(u, v) \,,$$

to compare with baselines LDP and trusted aggregator by enforcing $\overline{\varepsilon} = \max_{v \in \mathcal{V}} \overline{\varepsilon}_v \leq \varepsilon$. The value $\overline{\varepsilon}_v$ 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.

6.3. 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 [BGPS06], and with Erdös-Rényi graphs.

6.3.1. 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. Writing $W_{0:t} = W_{t-1} \dots W_0$, the iterates of $\mathcal{A}^T$ are thus defined by:

$$\forall v \in \mathcal{V}, x_v^0 = x_v + \eta_v \text{ with } \eta_v \sim \mathcal{N}(0, \sigma^2), \quad \text{and } x^{t+1} = W_t x^t = W_{0:t+1}(x + \eta) \,. \tag{6.4}$$

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:

$$\mathcal{O}_v\big(\mathcal{A}^T(\mathcal{D})\big) = \big\{ \big(W_{0:t}(x + \eta)\big)_w \,|\quad \{v, w\} \in \mathcal{E}_t \,, \quad 0 \leq t \leq T - 1 \big\} \cup \{x_v\} \,. \tag{6.5}$$

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 $\{v, w\}$. Under Assumption 6.2.1, $\mathcal{A}^T$ is (α, f)-PNDP with:*

$$f(u, v) = \frac{\alpha \Delta^2}{2\sigma^2} \sum_{w \in \mathcal{V}} \sum_{t \in \mathcal{P}^T_{\{v,w\}}} \frac{(W_{0:t})^2_{u,w}}{\|(W_{0:t})_w\|^2} \,. \tag{6.6}$$

Proof of Theorem 6.1. We need to bound the privacy loss that occurs from the following view:

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

We have:

$$D_\alpha\big(\mathcal{O}_v(\mathcal{A}^T(\mathcal{D})) \,||\, \mathcal{O}_v(\mathcal{A}^T(\mathcal{D}'))\big) \leq \sum_{t=0}^{T-1} \sum_{w \in \mathcal{N}_t(v)} D_\alpha\big(\big(W_{0:t}(x + \eta)\big)_w \,||\, \big(W_{0:t}(x' + \eta)\big)_w\big) \,.$$

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 $|(W_{0:t}(x'))_w - (W_{0:t}(x))_w|^2 \leq \Delta^2 (W_{0:t})^2_{u,w}$ under Assumption 6.2.1

and $\mathcal{D} \sim_u \mathcal{D}'$. Thus, using Lemma 6.2.1, we have:

$$D_\alpha\big(\big(W_{0:t}(x+\eta)\big)_w \,||\, \big(W_{0:t}(x'+\eta)\big)_w\big) \leq \frac{\alpha\Delta^2}{2\sigma^2}\frac{(W_{0:t})_{u,w}^2}{\|(W_{0:t})_w\|^2}\,,$$

leading to the desired $f(u,v)$. The mean privacy loss is then obtained by summing the above inequality for $u \neq v$ and $t < T$:

$$\begin{aligned}
\bar{\varepsilon}_v = \frac{1}{n}\sum_{u\neq v} f(u,v) &\leq \frac{1}{n}\sum_{u\in\mathcal{V}}\frac{\alpha\Delta^2}{2\sigma^2}\sum_{w\in\mathcal{V}}\sum_{t\in\mathcal{P}^T_{\{v,w\}}}\frac{(W_{0:t})_{u,w}^2}{\|(W_{0:t})_w\|^2} \\
&= \frac{1}{n}\frac{\alpha\Delta^2}{2\sigma^2}\sum_{t\in\mathcal{P}^T_{\{v,w\}}}\sum_{w\in\mathcal{V}}\sum_{u\in\mathcal{V}}\frac{(W_{0:t})_{u,w}^2}{\|(W_{0:t})_w\|^2} \\
&= \frac{1}{n}\frac{\alpha\Delta^2}{2\sigma^2}\sum_{t\in\mathcal{P}^T_{\{v,w\}}}\sum_{w\in\mathcal{V}} 1 \\
&= \frac{\alpha\Delta^2 T_v}{2n\sigma^2}\,,
\end{aligned}$$

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. □

This theorem gives a tight computation of the privacy loss between every pair of nodes and can easily be computed numerically (see Section 6.5). Since distant nodes correspond to small entries in $W_{0:t}$, Equation 6.6 suggests that they reveal less to each other. We will characterize this precisely for the case of fixed communication graph in the next subsection.

Another way to interpret the result of Theorem 6.1 is to derive the corresponding mean privacy loss:

$$\bar{\varepsilon}_v = \frac{\alpha\Delta^2 T_v}{2n\sigma^2}\,, \tag{6.7}$$

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.

6.3.2. Private Synchronous Muffliato

We now consider *Muffliato* over a fixed graph (Algorithm 6.1). Note that we use gossip acceleration (see Definition 6.2.3). 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.

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

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

Proof of Theorem 6.2. The desired result is a direct consequence of the following convergence bound.

Proposition 6.3.1 (Utility analysis)**.** *For any $T \geq 0$, the iterates $(x^T)_{T\geq 0}$ of* Muffliato *(Algo-*

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 σ^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

    $x_v^{t+1} \leftarrow (1-\gamma)x_v^{t-1} + \gamma\sum_{w\in\mathcal{N}_v} W_{v,w}x_w^t$

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 σ^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 $v \in \mathcal{V}\setminus\{v_t,w_t\}$, $x_v^{t+1} = x_v^t$

rithm 6.1) verify, for λ_W defined in Definition 6.2.2 and $\bar{x} = \frac{1}{n}\sum_{v\in\mathcal{V}} x_v \in \mathbb{R}^D$:

$$\frac{1}{2n}\sum_{v\in\mathcal{V}}\mathbb{E}\left[\left\|x_v^T - \bar{x}\right\|^2\right] \le \left(\frac{1}{n}\sum_{v\in\mathcal{V}}\|x_v-\bar{x}\|^2 + D\sigma^2\right)e^{-T\sqrt{\lambda_W}} + \frac{D\sigma^2}{n}\,. \tag{6.8}$$

Proof of Proposition 6.3.1. For $t \ge 0$ and $y \in \mathbb{R}^{\mathcal{V}\times D}$, using results from [BBG20b], we have, for a vector $y \in \mathbb{R}^{\mathcal{V}\times D}$ such that $\sum_{v\in\mathcal{V}} y_v = 0$, $\|P_t(W)y\|^2 \le 2(1-\sqrt{\lambda_W})^2\|y\|^2$. In particular:

$$\left\|P_t(W)(y-\bar{y}\mathbb{1}^\top)\right\| \le 2(1-\sqrt{\lambda_W})^t\left\|y-\bar{y}\mathbb{1}^\top\right\|^2,$$

where $\mathbb{1}$ is the vector with all entries equal to 1. Since

$$x^t = P_t(W)\big(x + \mathcal{N}(0,\sigma^2 I_{\mathcal{V}\times D})\big)\,,\quad t\ge 0\,,$$

we obtain that, for $\eta \sim \mathcal{N}(0,\sigma^2 I_{\mathcal{V}\times D})$ and $\bar{\eta} = \frac{1}{n}\sum_{v\in\mathcal{V}}\eta_v\mathbb{1}^\top \in \mathbb{R}^{\mathcal{V}\times D}$, using bias-variance decomposition twice:

$$\begin{aligned}
\frac{1}{2}\mathbb{E}\left[\left\|x^t-\bar{x}\right\|^2\right] &= \frac{1}{2}\mathbb{E}\left[\left\|P_t(W)(x^{(0)}-\bar{x})\right\|^2\right]\\
&= \frac{1}{2}\mathbb{E}\left[\|P_t(W)(x+\eta-\bar{x}-\bar{\eta})\|^2\right] + \frac{1}{2}\mathbb{E}\left[\|P_t(W)\bar{\eta}\|^2\right]\\
&\le (1-\sqrt{\lambda_W})^t\mathbb{E}\left[\|x+\eta-\bar{x}-\bar{\eta}\|^2\right] + \frac{D\sigma^2}{2n}\\
&\le (1-\sqrt{\lambda_W})^t\left(\mathbb{E}\left[\|x-\bar{x}\|^2\right] + nD\sigma^2\right) + \frac{D\sigma^2}{2n}\,.\square
\end{aligned}$$

The precision $\frac{3D\sigma^2}{n}$ is thus reached for

$$T^{\text{stop}}\big(W,(x_v)_{v\in\mathcal{V}},D\sigma^2\big) \le \sqrt{\lambda_W}^{-1}\ln\left(\frac{n}{D\sigma^2}\max\left(D\sigma^2,\frac{1}{n}\sum_{v\in\mathcal{V}}\|x_v-\bar{x}\|^2\right)\right).$$

Table 6.1 – *Utility of* Muffliato *for several topologies under the constraint* $\bar{\varepsilon} \leq \varepsilon$ *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	$\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n^2\varepsilon\lambda_W}\right)$	$\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n^2\varepsilon}\right)$	$\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n^{2-2/C}\varepsilon}\right)$	$\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n^2\varepsilon}\right)$	$\tilde{\mathcal{O}}\left(\frac{\alpha\Delta^2}{n\varepsilon}\right)$

□

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 6.3.1. *Algorithm 6.1 satisfies* $(\alpha, \varepsilon^T_{u\to v}(\alpha))$*-PNDP for node* u *with respect to* v*, with:*

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

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 $\bar{\varepsilon} = \max_{v\in\mathcal{V}} \bar{\varepsilon}_v \leq \varepsilon$ in Table 6.1 for various graphs, 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 [YYC+21], are also the ones that achieve optimal privacy amplification (up to logarithmic factors). In other words, *privacy, utility and scalability are compatible.*

6.3.3. 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 by [BGPS06, EBB+21]. 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 6.3 (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* $t \geq T^{\text{stop}}$*:*

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

Proof of Theorem 6.3. As in the synchronous case, we prove a more general convergence result that holds for D-dimensional inputs.

Proposition 6.3.2 (Utility analysis)**.** *For any $T \geq 0$, the iterates $(x^T)_{T\geq 0}$ of randomized* Muffliato *(Algorithm 6.2) verify:*

$$\frac{1}{2n}\sum_{v\in\mathcal{V}}\mathbb{E}\left[\left\|x_v^T-\bar{x}\right\|^2\right] \leq \left(\frac{1}{n}\sum_{v\in\mathcal{V}}\left\|x_v^0-\bar{x}\right\|^2 + D\sigma^2\right)e^{-T\lambda_2(p)} + \frac{D\sigma^2}{n}\,. \tag{6.9}$$

Proof of Proposition 6.3. For $t \geq 0$ and $y \in \mathbb{R}^{\mathcal{V}\times D}$, using results from [BGPS06], we have:

$$\mathbb{E}\left[\left\|W(t)(y-\bar{y}\mathbb{1}^\top)\right\|^2\right] \leq (1-\lambda(p))^t\left\|y-\bar{y}\mathbb{1}^\top\right\|^2,$$

where $\mathbb{1}$ with the vector with all entries equal to 1 and $\bar{y} = \frac{1}{n}\sum_{v\in\mathcal{V}} y_v$. 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

$$T^{\mathrm{stop}}\big(W,(x_v)_{v\in\mathcal{V}},\sigma^2\big) \leq \lambda(p)^{-1}\ln\left(\frac{n}{D\sigma^2}\max\left(D\sigma^2,\frac{1}{n}\sum_{v\in\mathcal{V}}\|x_v-\bar{x}\|^2\right)\right)\,. \tag{6.10}$$

□

In terms of privacy, randomized *Muffliato* satisfies the following guarantees, obtained by applying Theorem 6.1.

Corollary 6.3.2. *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,π_v) where $\pi_v = \sum_{w\sim v} p_{\{v,w\}}$. 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_vT communications. Consequently, as reported in Table 6.1, for a fixed privacy mean $\bar{\varepsilon}_v$, 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 λ_W as we do not accelerate randomized gossip (see Remark 6.3.1 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 6.3.1 (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 λ_W. 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 λ_W. However, Algorithm 6.2 can be accelerated using the continuized version of Nesterov acceleration we developed in Chapter 2 [EHM20, EBB+21, see also], thus replacing $\lambda(p)$ in the expression of T^{stop} in Theorem 6.3, 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.

6.3.4. 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 $q = c\ln(n)/n$ where $c > 1$, Erdös-Rényi graphs are good expanders with node degrees $d_v = \mathcal{O}(\log n)$ and λ_W concentrating around 1 [HKP19]. We obtain the following privacy guarantees.

Theorem 6.4 (*Muffliato* on a random Erdös-Rényi graph)**.** *Let $\alpha > 1$, $T \geq 0$, $\sigma^2 \geq \frac{\Delta^2\alpha(\alpha-1)}{2}$ and $q = c\frac{\ln(n)}{n}$ for $c > 1$. Let $u, v \in \mathcal{V}$ be distinct nodes. After running Algorithm 6.1 with these parameters, node u is $(\alpha, \varepsilon^T_{u\to v}(\alpha))$-PNDP with respect to v, with:*

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

Proof of Theorem 6.4. Theorem 6.4 is a consequence of a more general result, that we instantiate below (Theorem 6.5), before introducing notations and the class of random graphs we consider.

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 6.4 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 6.5 (*Muffliato* with a random graph)**.** *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 of Theorem 6.5. 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 $x^{(0)}_u + \mathcal{N}(0, \sigma^2)$ 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 for-

mula (6.6), 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 6.A.1 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:

$$\frac{1}{n}\left(\sum_{w\in\mathcal{N}(v)}\frac{\alpha\Delta^2}{2\sigma^2}+(n-d_v)\varepsilon^T_{u\to v}\right)\leq\frac{\alpha\Delta^2 d_v T}{n\sigma^2}\,.$$

Rearranging this inequality gives the result. □

□

This result 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.

6.4. 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

$$\phi(\theta)=\frac{1}{n}\sum_{v\in\mathcal{V}}\phi_v(\theta)\,,\quad\text{with}\quad\phi_v(\theta)=\frac{1}{|\mathcal{D}_v|}\sum_{x_v\in\mathcal{D}_v}\ell_v(\theta,x_v)\,,\quad\theta\in\mathbb{R}^D\,,\tag{6.11}$$

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 ϕ is μ-strongly convex, and each ϕ_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 ϕ, 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:

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

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 [KLB+20]. Inspired by the optimal algorithm MSDA of [SBB+17] that

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 σ^2, gossip matrices $(W_t)_{t\geq 0}$, local functions ϕ_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 $x_v^t \sim \mathcal{L}_v$
  $\theta_v^{t+1} = \text{MUFFLIATO}\big((\hat{\theta}_v^t)_{v\in\mathcal{V}}, W_t, K, \nu^2\sigma^2\big)$

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

Remark 6.4.1. *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 6.6 (Utility analysis of Algorithm 6.3)**.** *For suitable step-size parameters, for a total number of* T^{stop} *computations and* $T^{\text{stop}}K$ *communications, with:*

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

the iterates $(\theta^t)_{t\geq 0}$ *generated by Algorithm 6.3 verify* $\mathbb{E}\left[\phi(\tilde{\theta}^{\text{out}}) - \phi(\theta^\star)\right] = \tilde{\mathcal{O}}(\frac{D\sigma^2+\bar{\rho}^2}{\mu n T^{\text{stop}}})$ *where* $\tilde{\theta}^{\text{out}} \in \mathbb{R}^D$ *is a weighted average of the* $\bar{\theta}^t = \frac{1}{n}\sum_{v\in\mathcal{V}} \theta_v^t$ *until* T^{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 6.7 (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:*

$$\varepsilon_{u\to v}^T(\alpha) \leq \frac{T\Delta_\phi^2 \alpha}{2\sigma^2} \sum_{k=0}^{K-1} \sum_{w:\{v,w\}\in\mathcal{E}} \frac{(W^k)_{u,w}^2}{\|(W^k)_w\|^2}\,. \tag{6.12}$$

Thus, for any $\varepsilon > 0$, *Algorithm 6.3 with* $T^{\text{stop}}(\kappa, \sigma^2, n)$ *steps and for* K *as in Theorem 6.6, there exists* f *such that the algorithm is* (α, f)*-pairwise network DP, with:*

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

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 6.4.1) is privacy independent. It is typically dominated by the first term, which corresponds to the utility

[2]This assumption can be replaced by the more general Assumption 6.C.1 given in Appendix 6.C.

loss due to privacy. Comparing Theorem 6.7 with the results for *Muffliato* (Table 6.1 in Section 6.3.2), the only difference lies in the factor $D\Delta_\phi^2/\mu$. Note that Δ_ϕ^2 plays the role of the sensitivity Δ^2, and D appears naturally due to considering D-dimensional parameters. On the other hand, μ is directly related to the complexity of the optimization problem through the condition number κ: 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 6.3.1 applies here. In particular, for expander graphs like the exponential graph of [YYC+21], the factor $d/\sqrt{\lambda_W}$ is constant. In this case, converting to standard (ϵ, δ)-DP gives $\tilde{\mathcal{O}}\big(\frac{D\Delta_\phi^2}{\mu n^2 \epsilon^2}\big)$, recovering the optimal privacy-utility trade-off of central DP [BST14, WYX17] up to logarithmic factors. Remarkably, we achieve this 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 $T^{\text{stop}}(\kappa, \sigma^2, n)Kn = \tilde{\mathcal{O}}(\kappa n)$.

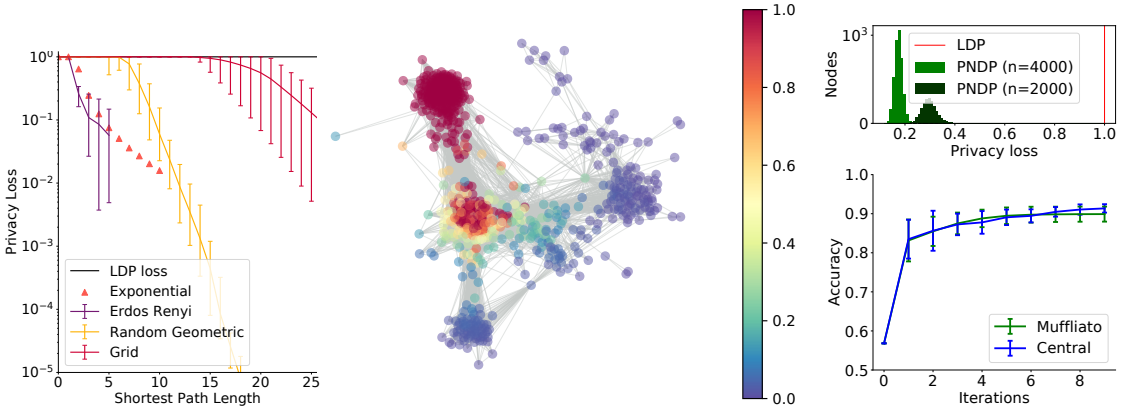
Remark 6.4.2 (Time-varying graphs)**.** *The analysis of* Muffliato*-GD/SGD presented in this section (Theorems 6.6 and 6.7) assumes constant gossip matrices $W_t = W$. A more general version of these results is presented in Appendix 6.C to handle time-varying matrices and graphs. This can be used to model randomized communications (as previously described for gossip averaging in Section 6.3.3) as well as user dropout (see experiments in Appendix 6.E). 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 6.5, where we randomize the graph after each gradient step of* Muffliato*-GD.*

6.5. 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-Renyi graphs, showing in each case clear gains in privacy compared to LDP.

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 privacy guarantees of *Muffliato* are deterministic for a fixed W, and defined by Equation 6.4. For each graph, we run *Muffliato* for the theoretical number of steps required for convergence, and report in Figure 6.1a 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 [YYC+21]. Consistently with our theoretical result, privacy is significantly amplified. The shortest path completely defines the privacy loss, so there is no variance. *Erdos-Renyi 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 6.4.2. *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 6.E.

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 uni-



(a) – *Synthetic graphs* (b) – *Facebook ego graph* (c) – *Logistic regression*

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.*

versity, 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 6.1b. We observe this consistently across other ego graphs (see Appendix 6.E). 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 6.11 with loss function $\ell(\theta; x, y) = \ln(1 + \exp(-y\theta^\top x))$ 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 $n = 2000$ and $n = 4000$ in Figure 6.1c (top). We see that, as discussed in Remark 6.4.2, 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 6.1c (bottom), both approaches behave similarly in terms of accuracy across iterations.

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:

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

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 [SBB+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 [EMS22a] that match the privacy loss variations, and to study privacy attacks [PRT22].

Appendix of Chapter 6

6.A. Preliminary Lemmas and Notations

For the privacy analysis when the graph is private and randomly sampled, we use the following result on the weak convexity of the Rényi divergence [FMTT18b].

Lemma 6.A.1 (Quasi-convexity of Rényi divergence [FMTT18b]). *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 ρ be a distribution over $\mathcal{I}$ and μ_ρ and ν_ρ be obtained by sampling i from ρ, and outputing a sample from μ_i and ν_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.

6.B. Proofs of Section 6.3

6.B.1. Privacy Analysis (Corollary 6.3.1)

Proof of Corollary 6.3.1. For a fixed gossip matrix W, we have $W_{0:t} = W^t$, so that Theorem 6.1 reads:

$$f(u,v) = \frac{\alpha\Delta^2}{2\sigma^2}\sum_{t<T}\sum_{w:\{v,w\}\in\mathcal{E}}\frac{(W^t)_{u,w}^2}{\|(W^t)_w\|^2}.$$

Since W is bi-stochastic, $\frac{(W^t)_{u,w}^2}{\|(W^t)_w\|^2} \leq n \times (W^t)_{u,w}^2 = n\mathbb{P}\left(X^t = u|X^0 = w\right)^2$, using Cauchy-Schwarz inequality.

Summing over the neighbors $w \sim v$, we obtain, for $t < T$:

$$\begin{aligned}
\sum_{w\sim v}\frac{\alpha}{2\sigma^2}\sum_{w:\{v,w\}\in\mathcal{E}}\frac{(W^t)_{u,w}^2}{\|(W^t)_w\|^2} &\leq \frac{\alpha n}{2\sigma^2}\sum_{w\sim v}\mathbb{P}\left(X^t = u|X^0 = w\right)^2 \\
&\leq \frac{\alpha n}{2\sigma^2}\left(\sum_{w\sim v}\mathbb{P}\left(X^t = u|X^0 = w\right)\right)^2 \\
&\leq \frac{\alpha n}{2\sigma^2}\frac{1}{\min_{w\sim v}W_{v,w}^2}\left(\sum_{w\sim v}W_{v,w}\mathbb{P}\left(X^t = u|X^0 = w\right)\right)^2 \\
&\leq \frac{\alpha n}{2\sigma^2}\frac{1}{\min_{w\sim v}W_{v,w}^2}\mathbb{P}\left(X^{t+1} = u|X^0 = v\right)^2,
\end{aligned}$$

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 6.3.1. □

6.B.2. 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 $T^{\mathrm{stop}} = \lambda_W^{-1/2} \ln\left(\frac{n}{\sigma^2}\max\left(\sigma^2, \frac{1}{n}\sum_{v\in\mathcal{V}}\left\|x_v^0 - \bar{x}\right\|^2\right)\right)$ 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 [MW89].

6.B.3. Second line of Table 6.1

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 $C > 0$,

$$\mathbb{P}\left(T_v - \mathbb{E}T_v \geq C\right) \leq \exp\Big(-\frac{C^2}{T}\Big)\,,$$

using Hoeffding's inequality. We take as time-horizon $T = T^{\mathrm{stop}} \geq 1/\lambda(p)$ defined in Theorem 6.3, leading to

$$\mathbb{P}\left(T_v - \mathbb{E}T_v \geq C\right) \leq \exp\Big(-\frac{C^2\lambda_W}{n}\Big)\,, \quad \mathbb{E}\left[\bar{\varepsilon}_v\right] = \tilde{\mathcal{O}}\Big(\frac{\alpha\Delta^2}{2n\sigma^2\lambda_W}\Big)\,,$$

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^{stop}, deriving σ^2 from this and thus the resulting utility) leads to the second line of Table 6.1.

6.C. 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].

6.C.1. Proof of Theorem 6.6 (Utility Analysis)

As before, we have a more general convergence result.

Theorem 6.8 (Utility analysis of Algorithm 6.3). *Let* $K \geq \left\lceil \sqrt{\lambda}^{-1} \ln\left(\max\left(n, \frac{\bar{\zeta}^2}{D\sigma^2+\bar{\rho}^2}\right)\right)\right\rceil$. *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 6.8 is a direct consequence of Theorem 2 in [KLB+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 $1 - p = 2(1 - \sqrt{\lambda})^K \leq 2\min(\frac{1}{n}, \frac{D\sigma^2+\bar{\rho}^2}{\bar{\zeta}^2})$.

6.C.2. Proof of Theorem 6.7 (Privacy Analysis)

The function Lipschitzness can in fact be replaced by a more general assumption.

Assumption 6.C.1. *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 6.9 (Privacy analysis of Algorithm 6.3)**.** *Let* $(W_t)_{0\leq t<T}$ *be a sequence of gossip matrices of spectral gap larger than* λ. *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:*

$$\varepsilon_{u\to v}^T(\alpha) \leq \frac{\Delta_\phi^2 \alpha}{2\sigma^2} \sum_{t=0}^{T-1} \sum_{k=0}^{K-1} \sum_{w:\{v,w\}\in\mathcal{E}_t} \frac{(W_t^k)_{u,w}^2}{\left\|(W_t^k)_w\right\|^2} \,. \tag{6.13}$$

Thus, for any $\varepsilon > 0$, *Algorithm 6.3 with* $T^{\mathrm{stop}}(\kappa, \sigma^2, n)$ *steps and for* K *as in Theorem 6.6, there exists* f *such that the algorithm is* (α, 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 6.9. 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. Using Theorem 6.1 (with Assumption 6.2.1 satisfied using Assumption 6.C.1) and a post processing inequality, round of communication t leads to a privacy leak of:

$$\frac{\Delta_\phi^2 \alpha}{2\sigma^2} \sum_{k=0}^{K-1} \sum_{w:\{v,w\}\in\mathcal{E}_t} \frac{(W_t^k)_{u,w}^2}{\left\|(W_t^k)_w\right\|^2} \,,$$

where $\mathcal{E}_t$ are the edges of the graph drawn at time t. We obtain the first inequality of Theorem 6.7 by summing this over $t < T$.

For the second inequality, we have, by summing:

$$\bar{\varepsilon}_v = \frac{1}{n}\sum_{u\neq v} \varepsilon_{u\to v}^T(\alpha) \leq \frac{KT\bar{d}_v \Delta_\phi^2 \alpha}{2n\sigma^2} \leq \frac{KT\bar{d}\Delta_\phi^2 \alpha}{2n\sigma^2} \,.$$

In order to reach a precision $\frac{D\sigma^2+\bar{\rho}^2}{n}$, are required $T = \mathcal{O}(\kappa \ln(D\sigma^2/n))$ iterations. Using $K = \tilde{\mathcal{O}}(1/\sqrt{\lambda})$, we have:

$$\bar{\varepsilon}_v = \mathcal{O}\left(\frac{\bar{d}\Delta_\phi^2 \alpha}{2n\sigma^2} \kappa \sqrt{\lambda}^{-1} \ln(D\sigma^2/n)\right) \,.$$

Taking σ^2 such that $\frac{\bar{d}\Delta_\phi^2\alpha}{2n\sigma^2}\kappa\sqrt{\lambda}^{-1}\ln(\sigma^2/n) = \varepsilon$ yields the desired result. □

6.D. 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.

6.D.1. Presence of Colluding Nodes

Definitions The notions of pairwise network DP we introduced in Section 6.2.3 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:

$$\mathcal{O}_V\big(\mathcal{A}(\mathcal{D})\big) = \bigcup_{v\in V} \mathcal{O}_V\big(\mathcal{A}(\mathcal{D})\big)\,,$$

or equivalently as:

$$\mathcal{O}_V\big(\mathcal{A}(\mathcal{D})\big) = \{(u, m(t), v) \in \mathcal{A}(\mathcal{D}) \quad \text{such that} \quad \{u,v\} \in \mathcal{E}\,, v \in V\}\,.$$

Below, $\mathcal{P}(\mathcal{V})$ denotes the powerset of $\mathcal{V}$.

Definition 6.D.1 (Pairwise Network DP with colluders)**.** *For $f : \mathcal{V}\times\mathcal{P}(\mathcal{V}) \to \mathbb{R}^+$, an algorithm $\mathcal{A}$ satisfies (α, 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:*

$$D_\alpha\big(\mathcal{O}_V(\mathcal{A}(\mathcal{D})) \,||\, \mathcal{O}_V(\mathcal{A}(\mathcal{D}'))\big) \leq f(u, V)\,. \tag{6.14}$$

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 (6.14) holds for $f(u,V)$. Finally, if for a function $f : \mathcal{V}\times\mathcal{P}(\mathcal{V}) \to \mathbb{R}$, inequality (6.14) holds for all $(u,V) \in \mathcal{V}\times\mathcal{P}(\mathcal{V})$ such that $u \notin V$, we say that $\mathcal{A}$ is (α, 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 6.10. *Assume that Assumption 6.2.1 holds. 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 (α, f)-PNDP with::*

$$f(u,V) \leq \frac{\alpha\Delta^2}{2\sigma^2} \sum_{w\in\mathcal{V}} \sum_{t\in\mathcal{P}^T_{\{V,w\}}} \frac{(W_{0:t})^2_{u,w}}{\|(W_{0:t})_w\|^2}\,. \tag{6.15}$$

Consequently, there exists f such that $\mathcal{A}^T$ is (α, f)-PNDP with:

$$\bar{\varepsilon}_V = \frac{1}{n} \sum_{u \in \mathcal{V} \setminus V} f(u, V) \leq \frac{\alpha \Delta^2}{2n\sigma^2} \sum_{t<T} |\mathcal{N}_t(V)| \,, \tag{6.16}$$

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 6.D.1. *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 6.D.2. *There exists $f : \mathcal{V} \times \mathcal{P}(\mathcal{V}) \to \mathbb{R}^+$ such that Algorithm 6.1 after T steps is (α, f)-PNDP with the following privacy-utility guarantees for any $V \subset \mathcal{V}$:*

$$\bar{\varepsilon}_V = \frac{1}{n} \sum_{u \in \mathcal{V} \setminus V} f(u, V) \leq \varepsilon \quad , \qquad \frac{1}{2n} \sum_{v \in \mathcal{V}} \mathbb{E}\left[\left\|x_v^{\text{out}} - \bar{x}\right\|^2\right] \leq \tilde{\mathcal{O}}\left(\alpha \frac{d_V \Delta^2}{\varepsilon n^2 \sqrt{\lambda_W}}\right) \,,$$

where x^{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 ε. 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 6.D.3 (*Muffliato* on a random graph with collusions)**.** *Let $\alpha > 1$, $T \geq 0$, $\sigma^2 \geq \frac{\Delta^2 \alpha(\alpha-1)}{2}$ and $q = c\frac{\ln(n)}{n}$ 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 6.4, node u is $(\alpha, \varepsilon^T_{u \to v}(\alpha))$-PNDP with respect to colluders V, with:*

$$\varepsilon^T_{u \to V}(\alpha) \leq \begin{cases} \dfrac{\alpha}{2\sigma^2} & \text{with probability } 1 - (1-q)^{|V|} \\ \dfrac{\alpha}{\sigma^2} \dfrac{T d_V}{n - d_V} & \text{with probability } (1-q)^{|V|} \end{cases} .$$

Compensating for Collusions with Time-Varying Graph Sampling We now consider the decentralized optimization problem of Section 6.4 in the presence of colluders, and analyze its privacy with time-varying graph sampling as in Appendix 6.C.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 κ), 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 $n, q = \frac{c \ln(n)}{n}$ 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 λ of order 1 [HKP19]). In this context, we have the following result.

Proposition 6.D.1. *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 6.7, node u is $(\alpha, \varepsilon^T_{u\to v}(\alpha))$-PNDP with respect to colluders V, with:*

$$\varepsilon^T_{u\to V}(\alpha) \leq \frac{\Delta_\phi^2 \alpha}{\sigma^2} \sum_{t=0}^{T^{\text{stop}}} \beta_t + (1-\beta_t)\frac{K|\mathcal{N}_t(V)|}{n - |\mathcal{N}_t(V)|}\,,$$

where $T^{\text{stop}} = \tilde{\mathcal{O}}(\kappa)$ and $K = \tilde{\mathcal{O}}(1/\sqrt{\lambda})$ (see Theorem 6.6), $(\beta_t)_t$ are i.i.d. *Bernoulli random variables of parameter $\mathbb{P}\left(\exists v \in \mathcal{V}\,,\, \{u,v\} \in \mathcal{E}_t\right) = 1-(1-q)^{|V|}$, and $|\mathcal{N}_t(V)|$ is the number of neighbors of V in the graph sampled at iteration t, of order $\frac{c|V|\ln(n)}{n}$.*

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 6.15.

6.D.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 [DR14, see e.g.,] 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 6.D.2 (Group Pairwise Network DP)**.** *For $f : \mathcal{V} \times \mathcal{P}(\mathcal{V}) \to \mathbb{R}^+$, an algorithm $\mathcal{A}$ satisfies (α, 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:*

$$D_\alpha\big(\mathcal{O}_v(\mathcal{A}(\mathcal{D}))\,||\,\mathcal{O}_v(\mathcal{A}(\mathcal{D}'))\big) \leq f(U,v)\,. \tag{6.17}$$

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 6.11. *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 $\{v,w\}$. Under Assumption 6.2.1, $\mathcal{A}^T$ is (α, f)-group PNDP for with:*

$$f(U,v) = \frac{\alpha\Delta^2}{2\sigma^2} \sum_{w\in\mathcal{V}} \sum_{t\in\mathcal{P}^T_{\{v,w\}}} \frac{\sum_{u\in U}(W_{0:t})^2_{u,w}}{\|(W_{0:t})_w\|^2}\,. \tag{6.18}$$

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

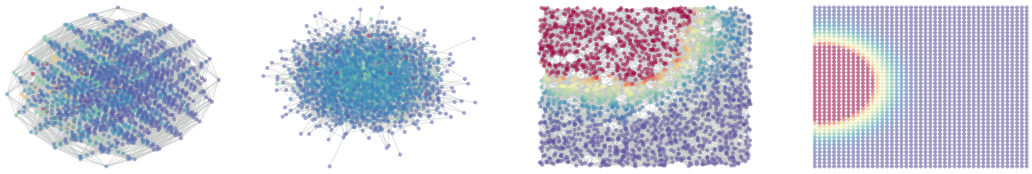


Figure 6.2 – *Level of the privacy loss for each node (color) with respect to a fixed node in the graph. These graphs corresponds to the graphs used in Figure 6.1a: from left to right, exponential graph, Erdos-Renyi graph, geometric random graph and grid.*

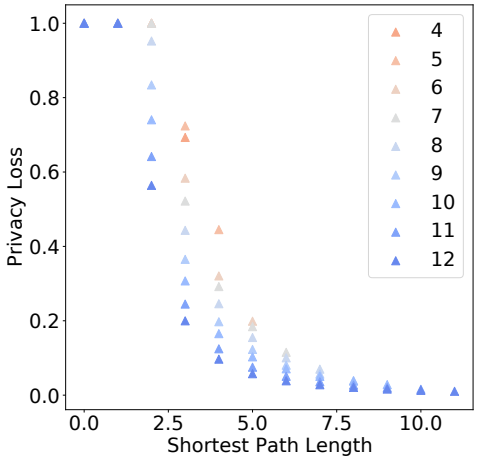


Figure 6.3 – *Privacy loss for the exponential graph with respect to the number of nodes n (following powers of 2).*

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.

6.E. Additional Numerical Experiments

6.E.1. Extra Synthetic graphs

Figure 6.1a 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 6.2.

We also report in Figure 6.3 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.

6.E.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

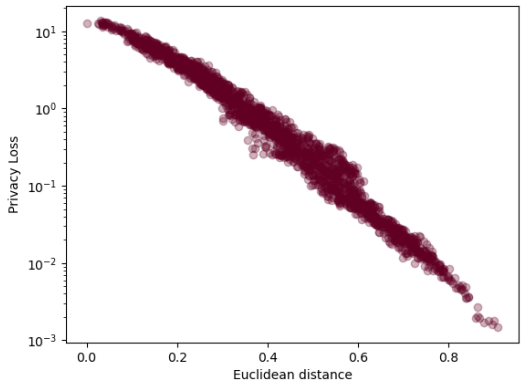


Figure 6.4 – *Privacy towards all the nodes as function of the Euclidean distance in a random geometric graph. We see a high level of correlation between the Euclidean distance and the privacy loss.*

prove that their privacy loss is the same, it is sufficient to exhibit an graph isomorphism Φ 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.

6.E.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.

Our experiments show that the privacy loss is extremely well correlated to the Euclidean distance, as represented in Figure 6.4. 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.

Figure 6.5 – *Privacy loss on the 9 other Facebook Ego graphs, following the same methodology as in Figure 6.1b.*

6.E.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.

6.E.5. Logistic Regression on Houses Dataset

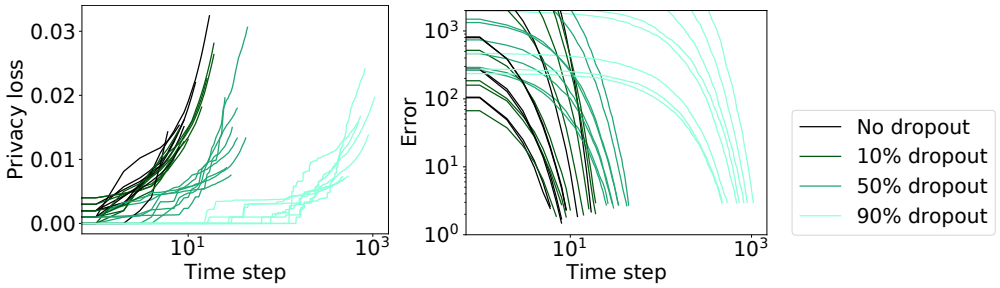
We report in Table 6.2 the parameters used in the experiments of Figure 6.1c.

6.E.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

Table 6.2 – *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



(a) – *Privacy dropout* (b) – *Error dropout*

Figure 6.6 – *Simulation (10 runs) of a dropout scenario (with different levels of dropout) with* $n = 1000$*, where at each gossip step an Erdös-Renyi 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.*

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 Renyi 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 6.6a, and report the privacy loss and its standard deviation at each dropout level in Table 6.3. 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.

Table 6.3 – *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	$(1.4 \pm 0.6) \times 10^{-2}$

Chapter 7

Stochastic gradient descent under Markovian sampling schemes

All the decentralized algorithms considered so far are gossip based decentralized algorithms or variations of gossip algorithms. Their advantage is that they allow to parallel computations, while performing underlying communications all over the graph. However, gossipping is not the only communication solution: instead, one might consider a random-walker in the graph – a *token* – that consists of the model to be updated. Once at some node location, the token updates the value of the model using the node's local data and when this is done, the updated model continues its random walk. Such algorithms have been used as privacy preserving algorithms and studied as such [CB20].

These token random-walk based algorithms differ from gossip algorithms, and are linked to a variation of vanilla stochastic gradient descent where the optimizer only has access to a "Markovian sampling scheme": *Markov chain SGD*. These schemes encompass applications that range from decentralized optimization with a random walker (*token algorithms*), to RL and online system identification problems.

This chapter focuses on obtaining rates of convergence for these methods under the least restrictive assumptions possible on the underlying Markov chain and on the functions optimized. We first unveil the theoretical lower bound for methods that sample stochastic gradients along the path of a Markov chain, making appear a dependency in the hitting time of the underlying Markov chain. We then study *Markov chain SGD* (MC-SGD) under much milder regularity assumptions than prior works. We finally introduce MC-SAG, an alternative to MC-SGD with variance reduction, that only depends on the hitting time of the Markov chain, therefore obtaining a communication-efficient token algorithm.

7.1. Introduction

In this chapter, as explained above, we consider a stochastic optimization problem that takes root in decentralized optimization, estimation problems, and Reinforcement Learning. Consider a function f defined as:

$$f(\mathbf{x}) = \mathbb{E}_{v\sim\pi}\left[f_v(\mathbf{x})\right], \quad \mathbf{x}\in\mathbb{R}^d, \tag{7.1}$$

where π is a probability distribution over a set $\mathcal{V}$, and f_v are smooth functions on $\mathbb{R}^d$ for all v in $\mathcal{V}$. Classicaly, this represents the loss of a model parameterized by $\mathbf{x}$ on data parameterized by v. If *i.i.d.* samples $(v_t)_{t\geq 0}$ of law π and their corresponding gradient estimates (∇f_{v_t}) were accessible, one could directly apply SGD-like algorithms, that have proved to be efficient in large scale machine learning problems [BCN18]. We however consider in this chapter a different setting: we assume the existence of a Markov chain (v_t) of state space $\mathcal{V}$ and stationary distribution π. The optimizer may then use biased stochastic gradients along the path of this Markov chain to perform incremental updates. She may for instance use the

Markov chain SGD (MC-SGD) algorithm, defined through the following recursion:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma \nabla f_{v_t}(\mathbf{x}_t)\,. \tag{7.2}$$

Being "ergodically unbiased", such iterates should behave closely to those of vanilla SGD. The analysis is however notoriously difficult, since in (7.2), variable $\mathbf{x}_t$ and the current state of the Markov chain v_t are not independent, so that $\mathbb{E}\left[\nabla f_{v_t}(\mathbf{x}_t)|\mathbf{x}_t\right]$ can be arbitrarily far from $\nabla f(\mathbf{x}_t)$. This chapter focuses on analyzing algorithms that incrementally sample stochastic gradients alongside the Markov chain (v_t), motivated by the following applications.

7.1.1. Token algorithms

Traditional machine learning optimization algorithms require data centralization, raising scalability and privavy issues, hence the alternative of Federated Learning, where users' data is held on device, and the training is orchestrated at a server level. Decentralized optimization goes further, by removing the dependency over a central entity, leading to increased scalability, privacy and robustness to node failures, broadening the range of applications to IoT (Internet of Things) networks. In decentralized optimization, users (or agents) are represented as nodes of a connected graph $G = (\mathcal{V}, \mathcal{E})$ over a finite set of users $\mathcal{V}$ (of cardinality n). The problem considered is then the minimization of

$$f(\mathbf{x}) = \frac{1}{n}\sum_{v\in\mathcal{V}} f_v(\mathbf{x})\,, \quad \mathbf{x}\in\mathbb{R}^d\,, \tag{7.3}$$

where each f_v is locally held by user $v \in \mathcal{V}$, using only communications between neighboring agents in the graph. There are several known decentralized algorithmic approaches to minimize f under these constrains. The prominent one consists in alternating between communications using gossip matrices [BGPS06, DKM+10] and local gradient computations, until a consensus is reached. These gossip approaches suffer from a high synchronization cost (nodes in the graph are required to perform simultaneous operations, or to be aware of operations at the other end of the communication graph) that can be prohibitive if we aim at removing the dependency on a centralized orchestrator. Further, a high number of communications are required to reach consensus, whether all nodes in the graph (as in synchronous gossip) or only two neighboring ones (as in randomized gossip) communicate at each iteration. To alleviate these communication burdens, based on the original works of [LS07, JRJ07, JRJ10], we study algorithms based on Markov chain SGD: a variable $\mathbf{x}$ performs a random walk on graph G, and is incrementally updated at each step of the random walk, using the local function available at its location. This approach thus boils down to the one presented above with the function defined in (7.1), where $\mathcal{V}$ is the (finite) set of agents, π is the uniform distribution over $\mathcal{V}$, and (v_t) is the Markov chain consisting of the consecutive states of the random walk performed on graph G. The random walk guarantees that every communications are spent on updating the global model, as opposed to gossip-based algorithms, where communications are used to reach a running consensus while locally performing gradient steps.

These algorithms are referred to as *token algorithms*: a token (that represents the model estimate) randomly walks the graph and performs updates during its walk. There are two directions to design and analyze token algorithms. [JRJ07] designed and analyzed its algorithm using, based on SGD with subdifferentials and a Markov chain sampling (consisting of the random walk). Following works [DAJJ11, SSY18] tried to improve convergence guarantees of such stochastic gradient algorithms with Markov chain sampling, under various scenarii (mirror SGD *e.g.*). However, all these analyses rely on overly strong assumption: bounded gradients and/or bounded domains are assumed, and the rates obtained are of the form $\tau_{\rm mix}/T + \sqrt{\tau_{\rm mix}/T}$ for a number T of steps, where $\tau_{\rm mix}$ is the mixing time of the underlying Markov chain. More recently, [DL22] obtained similar rates under similar assumptions (bounded losses and gradients), but without requiring any prior knowledge of $\tau_{\rm mix}$, using

adaptive stepsizes.

A more recent approach consists in deriving token algorithms from Lagrangian duality and from variants of coordinate gradient methods or ADMM algorithms with Markov chain sampling. [MYH+20] introduce the *Walkman* algorithm, whose analysis works on any graph, and obtain rates of $\frac{\tau_{\text{mix}}^2 n}{T}$ to reach approximate-stationary points, while [Hen22] introduced a more general framework, but whose analysis only works on the complete graph (and is thus equivalent to an *i.i.d.* sampling). The analysis of [Hen22] can however be extended to arbitrary graph, by performing gradient updates every τ_{mix} steps of the random walk (thus mimicking *i.i.d.*-ness), obtaining a a dependency on $\tau_{\text{mix}} n$, making their algorithm state of the art for these problems. Altenatively, [WLYZ22] studies the algorithm stability of MC-SGD in order to derive generalization upper-bounds for this algorithm, and [SLW22] provides and studies adaptive token algorithms. Recently, and concurrently to this work, [Doa23] also studies MC-SGD without smoothness; however, their dependency on the mixing time of the random walk (in their Theorem 1) scales as $\exp(c\tau_{\text{mix}})$: this is prohibitive as soon as the mixing time becomes larger than $\mathcal{O}(1)$.

In summary, current token algorithms and their analyses either rely on strong noise and regularity assumptions (*e.g.* bounded gradients), or suffer from an overly strong dependency on Markov chain-related quantities (as in [MYH+20, Hen22]).

The token algorithms we consider are to be put in contrast with *consensus-based decentralized* or *gossip* algorithms (with fixed gossip matrices [DKM+10] or with randomized pairwise communications [BGPS06]) considered in the previous chapters. They originally were introduced to compute the global average of local vectors through peer-to-peer communication. Among the classical decentralized optimization algorithms, some alternate between gossip communications and local steps [NO09, KSJ19, KLB+20], others use dual formulations and formulate the consensus constraint using gossip matrices to obtain decentralized dual or primal-dual algorithms [SBB+17, HBM19, EBB+21, KGGR21b, AS19], and benefit from natural privacy amplification mechanisms [CEBM22]. Other approaches include non-symmetric communication matrices [AR21] that are more scalable. We refer the reader to [NOR18] for a broader survey on decentralized optimization. The works we relate to in this line of research are [KLB+20], where a unified analysis of decentralized SGD is performed (the "gossip equivalent" of our algorithm MC-SGD), and in particular contains rates for convex-non-smooth functions, and [YJY19], that performs an analysis of decentralized SGD with momentum in the smooth-non-convex case, which is the "gossip equivalent" of our algorithm MC-SAG.

7.1.2. Reinforcement Learning problems and online system identification

In several applications (RL, time-series analysis *e.g.*), a statistician may have access to values $(X_t)_{t\geq 0}$ generated sequentially along the path of a Markov chain, observations from which she wishes to estimate a parameter For instance, [KNJN21] consider a sequence of observations $X_{t+1} = A_\star X_t + \xi_t$ for ξ_t *i.i.d.* centered noise, and $A_\star$ to estimate, and aim at finding $\hat{A}$ minimizing the MSE $\mathbb{E}\left[\left\|\sum_{t<T}(\hat{A} - A_\star)X\right\|\right]$ where $X \sim \pi$ is the stationary distribution. Studying this problem under the lens of stochastic optimization, this boils down to building efficient strategis for SGD under Markov chain sampling, beyond the case of linear mean-squared regressions studied in [KNJN21]. While optimal offline policies have extensively been studied in this setting [JP19, SMT+18], online algorithms that take the form of SGD-like algorithms have received little attention, and only focus on the case of quadratic losses and Markov chain as described above (*i.e.*, Markov chains of the form $X_{t+1} = A_\star X_t + \eta_t$). Under these specific assumptions, [NWB+20] prove that a dependency on τ_{mix} for MC-SGD is inevitable, while using reverse-experience replay, [KNJN21] obtains sample-optimal online algorithms. Their algorithm however require to store a number of iterates that grow linearly with τ_{mix}.

The convergence guarantees we prove in the sequel for SGD under Markov chain sam-

pling fit in this online framework, and refine previous analyses by removing strong regularity assumptions such as bounded iterates or bounded gradients [SSY18], or strong assumptions on the Markovian structure data and least-squares problems [NWB+20, KNJN21].

Finally, note that in our setting, the iterates of the algorithms considered (denoted as $(\mathbf{x}_k)_{k\geq 0}$) and the Markov chain $(v_k)_{k\geq 0}$ are dependent of each other. More precisely, $(v_k)_{k\geq 0}$ is a Markov chain whose states do not depend on the iterate sequence, while $\mathbf{x}_k$ is $(v_\ell)_{\ell\leq k-1}$-measurable. This setting is sometimes referred to as *exogenous Markov noise* [Rus86]. Another line of works, pioneered by [BMP90], considers Markov transitions for (v_k) where $v_{k+1}|v_k$ is sampled using a Markov transition kernel $P_{\mathbf{x}_0,\dots,\mathbf{x}_{k-1}}$ that is directly linked to the iterates. This orthogonal line of work of stochastic approximation with Markovian noise is related to sampling (through the MCMC algorithm), adaptive filtering, and other related problems that involve exploration [BR85, AMP05, AM06, FMSV16, BL23]. Our work aims at finding precise rates of convergence as in the convex or non-convex optimization literature [Bub15, CDHS21], under the mildest assumptions on the exogenous Markov-chain $(v_k)_{k\geq 0}$.

7.1.3. Outline of this chapter

In this chapter, we analyze theoretically stochastic gradient methods with Markov chain sampling (such as MC-SGD in Equation (7.2)), and aim at deriving complexity bounds under the mildest assumptions possible. We first derive in Section 7.3 complexity lower bounds for such methods, making appear τ_{hit} as the Markov chain quantity that slows down such algorithms.

We then study MC-SGD under various regularity assumptions in Section 7.4: we remove the bounded gradient assumption of all previous analyses, obtain rates under a μ-PL assumption, and prove a linear convergence in the interpolation regime, where noise and function dissimilarities only need to be bounded at the optimum.

In the data-heterogeneous setting (functions f_v that can be arbitrarily dissimilar) and in the case where $\mathcal{V}$ (the state space of the Markov chain) is finite, we introduce MC-SAG in Section 7.5, a variance-reduced alternative to MC-SGD, that is perfectly suited to decentralized optimization. Using time adaptive stepsizes, this algorithm has a rate of convergence of τ_{hit}/T and thus matches that of our lower bound, up to acceleration.

We discuss in Section 7.6 the implications of our results. In particular, we prove that random-walk based decentralization is more communication efficient than consensus-based approaches; prior to our analysis, this was only shown empirically [MYH+20, JRJ10]. Further, our results formally prove that using all gradients along the Markov chain trajectory leads to faster rates; as in the previous case, this was only empirically observed before [SSY18]. These two consequences are derived from the fact that MC-SAG depends only on τ_{hit} rather than the traditionally used quantity $n\tau_{\text{mix}}$, that can be arbitrarily bigger (Table 7.2).

7.2. Markov chains preliminaries

7.2.1. Definitions and notations

We refer the interested reader to [LPW06] for a thorough introduction to Markov chain theory. In this section, we define mixing, hitting and cover times for a Markov chain on a finite space set $\mathcal{V}$ of cardinality n. However, note that these definitions can be extended to the more general setting where $\mathcal{V}$ is infinite (either countable or not). In this chapter, all results that involve only the mixing time of the Markov chain (the results from Section 7.4) easily generalize to infinite state spaces.

Definition 7.2.1. *Let $P \in \mathbb{R}^{\mathcal{V}\times\mathcal{V}}$ be a stochastic matrix (*i.e. *$P_{v,w} \geq 0$ for all $v, w \in \mathcal{V}$ and $\sum_{w\in\mathcal{V}} P_{v,w} = 1$ for all $v \in \mathcal{V}$). A time-homogeneous Markov chain on $\mathcal{V}$ of transition matrix P is a stochastic process $(X_t)_{t\geq 0}$ with values in $\mathcal{V}$ such that, for any $t \geq 0$ and*

$w, v_0, \dots, v_{t-1}, v \in \mathcal{V}$,

$$\mathbb{P}\left(X_{t+1} = w | X_t = v, X_{t-1} = v_{t-1}, \dots, X_0 = v_0\right) = P_{v,w}\,.$$

A Markov chain of transition matrix P is irreducible if, for any $v, w \in \mathcal{V}$, there exists $t \geq 0$ such that $(P^t)_{vw} > 0$. A Markov chain of transition matrix P is aperiodic if there exists $t_0 > 0$ such that for all $t \geq t_0$ and $v, w \in \mathcal{V}$, $(P^t)_{v,w} > 0$. Any irreducible and aperiodic Markov chain on $\mathcal{V}$ admits a stationary distribution π, that verifies $\pi P = \pi$. It finally holds that, if P is reversible ($\pi_v P_{v,w} = \pi_w P_{w,v}$ for all $v, w \in \mathcal{V}$), denoting as $\lambda_P = 1 - \max_{\lambda \in \mathrm{Sp}(P)\setminus\{1\}} |\lambda| > 0$ the absolute spectral gap of P, where $\mathrm{Sp}(P)$ is the spectrum of P, for any stochastic vector $\pi_0 \in \mathbb{R}^{\mathcal{V}}$: [1]

$$\left\|\pi_0 P^t - \pi\right\|_\pi \leq (1 - \lambda_P)^t \|\pi_0 - \pi\|_\pi\,.$$

If the chain is not reversible, there is still a linear decay, but in terms of total variation distance rather than in the norm $\|\cdot\|_\pi$ (Chapter 4.3 of [LPW06]). In the sequel, $(v_t)_{t\geq 0}$ is any irreducible aperiodic Markov chain of transition matrix P on $\mathcal{V}$ of stationary distribution π (not necessarily the uniform distribution on $\mathcal{V}$).

Furthermore, we define the graph $G = (\mathcal{V}, \mathcal{E})$ over the state space $\mathcal{V}$ through the relation $\{v, w\} \in \mathcal{E} \iff P_{v,w} > 0$ for v and w two distinct states. Consequently, the Markov chain $(v_t)_t$ can also be seen as a random walk on graph G, with transition probability P. In the random walk decentralized optimization case, this graph coincides with the communication graph. In the sequel, for $t \geq 0$ and $v \in \mathcal{V}$, $\mathbb{E}\left[\cdot | v_t = v\right]$ and $\mathbb{P}\left(\cdot | v_t = v\right)$ respectively denote the expectation and probability conditioned on the event $v_t = v$. Similarly for π_t a probability distribution on $\mathcal{V}$, $\mathbb{E}\left[\cdot | v_t \sim \pi_t\right]$ and $\mathbb{P}\left(\cdot | v_t \sim \pi_t\right)$ refers to conditioning on the law of v_t.

Definition 7.2.2 (Mixing, hitting and cover times)**.** *For $w \in \mathcal{V}$, let $\tau_w = \inf\{t \geq 1 \,|\, v_t = w\}$ be the time the chain reaches w (or returns to w, in the case $v_0 = w$). We define the following quantities.*

1. Mixing time. *For $\varepsilon > 0$, the mixing time $\tau_{\mathrm{mix}}(\varepsilon)$ of (v_t) is defined as, where $\mathrm{d_{TV}}$ is the total-variation distance:*

$$\tau_{\mathrm{mix}}(\varepsilon) = \inf\left\{t \geq 1 \,|\, \forall \pi_0\,, \mathrm{d_{TV}}(P^t \pi_0, \pi) \leq \varepsilon\right\},$$

and we define the mixing time τ_{mix} as $\tau_{\mathrm{mix}} = \tau_{\mathrm{mix}}(\pi_{\min}/2)$, where $\pi_{\min}$ is the stationary distribution. [2] *where $\pi_{\min} = \min_{v\in\mathcal{V}} \pi_v$.*

2. Hitting and cover times. *The hitting time τ_{hit} and cover time τ_{cov} of (v_t) are defined as:*

$$\tau_{\mathrm{hit}} = \max_{(v,w)\in\mathcal{V}^2} \mathbb{E}\left[\tau_w | v_0 = v\right],$$

$$\tau_{\mathrm{cov}} = \max_{v\in\mathcal{V}} \mathbb{E}\left[\max_{w\in\mathcal{V}} \tau_w | v_0 = v\right].$$

The mixing time is the number of steps of the Markov chain required for the distribution of the current state to be close to the stationary probability π. Starting from any arbitrary v_0, the hitting time bounds the time it takes to reach any fixed w, while the cover time bounds the number of steps required to visit all the nodes in the graph.

Under reversibility assumptions, we defined the *relaxation time* of the Markov chain as $\tau_{\mathrm{rel}} = 1/\lambda_P$.

[1] we write $\|z\|_\pi^2 = \sum_{v\in\mathcal{V}} \pi_v z_v^2$ for $z \in \mathbb{R}^{\mathcal{V}}$, and $\|\cdot\|$ always stands for the Euclidean norm

[2] this definition of mixing time is not standard: [LPW06] define it as $\tau_{\mathrm{mix}}(1/4)$, [MYH+20] define it as we do; however, as explained in Chapter 4.5 of [LPW06], these definitions are equivalent up to a factor $\ln(1/\pi_{\min})$

7.2.2. Relation between mixing, hitting and cover times

We first begin by the two following lemmas, that compare the three quantities introduced above: mixing, hitting and cover times. The first one is very classical, and bounds the mixing time in terms of $1/\lambda_P$, in the case where the chain is reversible; we provide a proof for completeness. The second lemma we provide bounds the hitting time of the Markov chain with the mixing time. This result is somewhat less classical, and is not present in the classical Markov chain literature surveys.

Lemma 7.2.1 (τ_{mix} and λ_P)**.** *For any $\varepsilon > 0$, if the chain is reversible:*

$$\tau_{\text{mix}}(\varepsilon) \leq \left\lceil \frac{1}{\lambda_P} \ln(\varepsilon^{-1}\pi_{\min}^{-1}) \right\rceil ,$$

so that $\tau_{\text{mix}} \leq \left\lceil \frac{1}{\lambda_P} \ln(\pi_{\min}^{-2}/2) \right\rceil$.

Proof. We have:

$$\begin{aligned}
\mathrm{d}_{\text{TV}}(P^t\pi_0, \pi) &= \frac{1}{2}\sum_{w\in\mathcal{V}} |(P^t\pi_0)_w - \pi_w| \\
&\leq \frac{1}{2\pi_{\min}} \sum_{w\in\mathcal{V}} \pi_w |(P^t\pi_0)_w - \pi_w| \\
&\leq \frac{1}{2\pi_{\min}} \sqrt{\|P^t\pi_0 - \pi\|_\pi^2} \\
&\leq \frac{(1-\lambda_P)^t}{2\pi_{\min}} \|\pi_0 - \pi\|_\pi \\
&\leq \frac{(1-\lambda_P)^t}{\pi_{\min}} ,
\end{aligned}$$

so that $|(P^t)_{v,w} - \pi_w| \leq \varepsilon$ for $t \geq \lambda_P^{-1}\ln(\pi_{\min}^{-1}\varepsilon^{-1}/2)$. □

Lemma 7.2.2 (Mixing times and hitting times)**.**

$$\tau_{\text{hit}} \leq 2\pi_{\min}^{-1}\tau_{\text{mix}} ,$$

so that if π is the uniform distribution over $\mathcal{V}$, $\tau_{\text{hit}} \leq 2n\tau_{\text{mix}}$.

Proof. for any $v, w \in \mathcal{V}$,

$$\begin{aligned}
\mathbb{E}\left[\tau_w | v_0 = v\right] &= \sum_{k\geq 1} \mathbb{P}\left(\tau_w \geq k | v_0 = v\right) \\
&\leq \sum_{\ell \geq 0} \mathbb{P}\left(\tau_w > \ell\tau_{\text{mix}} | v_0 = v\right) .
\end{aligned}$$

Then, for $\ell \geq 0$,

$$\mathbb{P}\left(\tau_w > (\ell+1)\tau_{\text{mix}} | v_0 = v\right) = \mathbb{P}\left(\tau_w > (\ell+1)\tau_{\text{mix}} | \tau_w > \ell\tau_{\text{mix}}, v_0 = v\right) \mathbb{P}\left(\tau_w > \ell\tau_{\text{mix}} | v_0 = v\right) ,$$

and, conditioning on $v_{\ell\tau_{\text{mix}}}$, $\mathbb{P}\left(\tau_w > (\ell+1)\tau_{\text{mix}} | \tau_w > \ell\tau_{\text{mix}}, v_{\ell\tau_{\text{mix}}}\right) \leq \mathbb{P}\left(v_{(\ell+1)\tau\text{mix}} \neq w | v_{\ell\tau_{\text{mix}}}\right)$. By definition of τ_{mix}, we have that $\mathbb{P}\left(v_{(\ell+1)\tau\text{mix}} \neq w | v_{\ell\tau_{\text{mix}}}\right) \leq (1 - \pi_w/2)$, so that:

$$\mathbb{P}\left(\tau_w > (\ell+1)\tau_{\text{mix}} | v_0 = v\right) \leq (1-\pi_w/2)\mathbb{P}\left(\tau_w > \ell\tau_{\text{mix}} | v_0 = v\right) ,$$

and $\mathbb{P}\left(\tau_w > \ell\tau_{\text{mix}} | v_0 = v\right) \leq (1 - \pi_w/2)^\ell$ by recursion. Finally,

$$\begin{aligned}\mathbb{E}\left[\tau_w | v_0 = v\right] &\leq \tau_{\text{mix}} \sum_{\ell \geq 0} (1 - \pi_w/2)^\ell \\ &\leq \frac{2\tau_{\text{mix}}}{\pi_w},\end{aligned}$$

concluding the proof by taking the maximum over w. □

Matthews' bound for cover times The following result bounds the cover time of the Markov chain: it is in fact closely related to its hitting time, and the two differ with a most a factor $\ln(n)$. This surprising result is proved in a very elegant way in the survey [LPW06], using the famous Matthews' method [Mat88].

Theorem 7.1 (Matthews' bound for cover times)**.** *The hitting and cover times of the Markov chain verify:*

$$\tau_{\text{cov}} \leq \left(\sum_{k=1}^{n-1} \frac{1}{k}\right) \tau_{\text{hit}}\,.$$

A bound on the hitting time of regular and symetric graphs Using results from [Rao12], we relate the hitting time of symmetric regular graphs (in a sense that we define below) to well-known graph-related quantities: number of edges $|\mathcal{E}|$, diameter δ and degree d.

Lemma 7.2.3 (Bounding hitting times of regular graphs)**.** *Let (v_t) be the simple random walk on a d-regular graph G of diameter δ, that satisfies the following symetry property: for any $\{u, v\}, \{v, w\} \in \mathcal{E}$, there exists a graph automorphism that maps v to w. Then, we have:*

$$\tau_{\text{hit}} \leq \frac{2|\mathcal{E}|\delta}{d}$$

Proof. Using Theorem 2.1 of [Rao12], for $\{v, w\} \in \mathcal{E}$, we have

$$\mathbb{E}\left[\tau_w | v_0 = v\right] = \frac{2|\mathcal{E}|}{d}\,,$$

where $|\mathcal{E}|$ is the number of edges in the graph. Let v and w in $\mathcal{V}$, at distance $\delta' \leq \delta$. There exists nodes $v = v(0), v(1) \ldots, v(\delta' - 1), v(\delta') = w$ such that for all $0 \leq s < \delta'$, $\{v(s), v(s+1)\} \in \mathcal{E}$, and by using the Markov property:

$$\mathbb{E}\left[\tau_w | v_0 = v\right] \leq \sum_{s < \delta'} \mathbb{E}\left[\tau_{v(s+1)} v_0 = v(s)\right] \leq \delta \frac{2|\mathcal{E}|}{d}\,.$$

□

7.3. Oracle complexity lower bounds under Markov chain sampling

In this section, we provide oracle complexity lower bounds for finding stationary points of the function f defined in (7.3), for a class of algorithms that satisfy a "Markov sampling scheme". For a given Markov chain (v_t) on $\mathcal{V}$, we consider algorithms verifying the following procedural constraints, for some fixed initialization $\mathcal{M}_0 = \{\mathbf{x}_0\}$ an then for $t \geq 0$,

1. A iteration t, the algorithm has access to function f_{v_t} and may extend its memory:

$$\mathcal{M}_{t+1} = \text{Span}(\{\mathbf{x}\,,\, \nabla f_{v_t}(\mathbf{x})\,, \quad \mathbf{x} \in \mathcal{M}_t\})\,.$$

2. **Output:** the algorithm specifies an output value $\mathbf{x}_t \in \mathcal{M}_t$.

We call algorithms verifying such constraints "black box procedures with Markov sampling (v_t)". Such procedures as well as the result below are inspired by the distributed black-box procedures defined in [SBB+17]. We use the notation $a(\cdot) = \Omega(b(\cdot))$ for $\exists C > 0$ such that $a(\cdot) \geq Cb(\cdot)$ in the theorem below, and classically consider the limiting situation $d \to \infty$, by assuming we are working in $\ell_2 = \left\{(\theta_k)_{k\in\mathbb{N}} \in \mathbb{R}^{\mathbb{N}} : \sum_k \theta_k^2 < \infty\right\}$.

Theorem 7.2. *Assume that τ_v (see Definition 7.2.2) has finite second moment for any $v \in \mathcal{V}$. Let $\Delta, B > 0$, $L > 0$ and $\mu > 0$, denote $\kappa = L/\mu$. Let $\mathbf{x}_0$ be fixed.*

1. Non-convex lower bound: *there exist functions $(f_v)_{v\in\mathcal{V}}$ such that $f = \sum_{v\in\mathcal{V}} \pi_v f_v$ is L-smooth, and $f(\mathbf{x}_0) - \min_{\mathbf{x}} f(\mathbf{x}) \leq \Delta$ and such that for any T and any Markov black-box algorithm that outputs $\mathbf{x}_T$ after T steps, we have:*

$$\|\nabla f(\mathbf{x}_T)\|^2 = \Omega\left(L\Delta\left(\frac{\tau_{\text{hit}}}{T}\right)^2\right).$$

2. Convex lower bound: *there exist functions $(f_v)_{v\in\mathcal{V}}$ such that $f = \sum_{v\in\mathcal{V}} \pi_v f_v$ is convex and L-smooth and minimized at some $\mathbf{x}^\star$ that verifies $\left\|\mathbf{x}^0 - \mathbf{x}^\star\right\|^2 \leq B^2$, and such that for any T and any Markov black-box algorithm that outputs $\mathbf{x}_T$ after T steps, we have:*

$$f(\mathbf{x}_T) - f(\mathbf{x}^\star) = \Omega\left(LB^2\left(\frac{\tau_{\text{hit}}}{T}\right)^2\right).$$

3. Strongly convex lower bound: *there exist functions $(f_v)_{v\in\mathcal{V}}$ such that $f = \sum_{v\in\mathcal{V}} \pi_v f_v$ is μ-strongly convex and L-smooth and minimized at some $\mathbf{x}^\star$ that verifies $\left\|\mathbf{x}^0 - \mathbf{x}^\star\right\|^2 \leq B^2$, and such that for any T and any Markov black-box algorithm that outputs $\mathbf{x}^T$ after T steps, we have:*

$$f(\mathbf{x}_T) - f(\mathbf{x}^\star) = \Omega\left(LB^2\exp\left(-\frac{T}{\sqrt{\kappa}\tau_{\text{hit}}}\right)\right).$$

Proof overview, non-convex case. For $\mathbf{x} \in \ell_2$ and $k \in \mathbb{N}$, denote by $\mathbf{x}(k)$ its k^{th} coordinate.We split the function defined in Section 3.2 of [CDHS21] (inspired by the "most difficult function in the world" of [Nes14]) between two nodes $v, w \in \mathcal{V}$ maximizing $\mathbb{E}\left[\tau_w | v_0 = v\right]$, by setting $\pi_v f_v(\mathbf{x}) = \frac{1}{2}\sum_{k\geq 1} 2\mathbf{x}(2k)^2 - 2\mathbf{x}(2k-1)\mathbf{x}(2k) + \frac{1}{2}\alpha\mathbf{x}(0)^2 - b\mathbf{x}(0) + \frac{\alpha}{2}$ and $\pi_w f_w(\mathbf{x}) = \frac{1}{2}\sum_{k\geq 0} 2\mathbf{x}(2k+1)^2 - 2\mathbf{x}(2k+1)\mathbf{x}(2k)$ for some $b, \alpha > 0$. Then, defining $T_0 = \tau_v$ and for $t \geq 0$,

$$\begin{aligned} T_{2k+1} &= \inf\left\{t \geq T_{2k} \,|\, v_t = w\right\} \quad \text{and} \\ T_{2k+2} &= \inf\left\{t \geq T_{2k+1} \,|\, v_t = w\right\}, \end{aligned}$$

so that any black box procedure with Markov sampling v_t with $\mathbf{x}_0 = 0$ satisfies, for any $t \leq T_k$, $\mathbf{x}_t \in \text{Span}(\mathbf{e}_0, \ldots, \mathbf{e}_{k-1})$ (where $(\mathbf{e}_i)$ is the canonical basis of ℓ_2) for any $k \geq 0$. Consequently, for $t \leq T_k$, one has, using results from [CDHS21]:

$$\|\nabla f(\mathbf{x}_t)\|^2 \geq \frac{L\Delta}{16k^2}.$$

For any $t \geq 0$, there exists some $k(t) \geq 0$ such that $T_{k(t)} \leq t < T_{k(t)+1}$, yielding $\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] \geq \mathbb{E}\left[\frac{L\Delta}{16k(t)^2}\right] \geq \frac{L\Delta}{16\mathbb{E}[k(t)]^2}$ using Jensen inequality. Finally, we prove a bound of the form $\mathbb{E}k(t) = \mathcal{O}(t/\tau_{\text{hit}})$, concluding the proof. □

A complete proof can be found in Appendix 7.A. The hitting time of the Markov chain bounds the mean time it takes to reach any other state in the graph, starting from any point in $\mathcal{V}$. Making no other assumptions than smoothness, having rates that depend on this hitting time is thus quite intuitive.

7.4. Analysis of Markov-Chain SGD

We have shown in last subsection that, in order to reach an ε-stationary point with Markov sampling, the optimizer is slowed down by the hitting time of the Markov chain; this lower bound being worst-case on the functions (f_v), we here add additional similarity assumptions, that are still milder than classical ones in this setting [SSY18].Studying the iterates generated by (7.2), we obtain in this section a dependency on τ_{mix}, provided bounded gradient dissimilarities (Assumptions 7.4.1 and 7.4.3).

We here assume that $(v_t)_{t\geq 0}$ is a Markov chain on $\mathcal{V}$ of invariant probability π (not necessarily the uniform measure on $\mathcal{V}$). Our analysis of Markov chain SGD **does not** rely on finite state spaces: $\mathcal{V}$ is not assumed to be finite (it can be any infinite countable, or continuous space). In this section, the function f studied is defined as

$$f(\cdot) = \mathbb{E}_{v\sim\pi}[f_v(\cdot)]\,,$$

as in (7.1). Consequently, for the MC-SGD algorithm for decentralized optimization over a given graph G to minimize the averaged function over all nodes (as in (7.3)), π needs to be the uniform probability over $\mathcal{V}$.

We first derive convergence rates under smoothness assumptions with or without a μ-PL inequality that holds, before improving our results under strong convexity assumptions, under which we prove a linear convergence rate in the interpolation regime. We finally add local noise (due to sampling, or additive gaussian noise to enforce privacy) in the final paragraph of this Section.

7.4.1. Analysis under bounded gradient dissimilarities

Assumption 7.4.1. *There exists $(\sigma_v^2)_{v\in\mathcal{V}}$ such that for all $v\in\mathcal{V}$ and all $\mathbf{x}\in\mathbb{R}^d$, we have*[3]*:*

$$\|\nabla f_v(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \sigma_v^2\,,$$

and we denote $\bar{\sigma}^2 = \mathbb{E}_{v\sim\pi}\left[\sigma_v^2\right]$ and $\sigma_{\max}^2 = \max_{v\in\mathcal{V}}\sigma_v^2$.

Assumption 7.4.2. *Each f_v is L-smooth, f is lower bounded, its minimum is attained at some $\mathbf{x}^\star\in\mathbb{R}^d$.*

Theorem 7.3 (MC-SGD)**.** *Assume that Assumptions 7.4.1 and 7.4.2 hold, and let $\Delta \geq f(\mathbf{x}_0) - f(\mathbf{x}^\star) + \sigma_{\max}^2/L$.*

1. *For a constant time-horizon dependent step size γ (*i.e.*, γ is a functiion of T), the iterates generated by Equation (7.2) satisfy, for $T\geq 2\tau_{\text{mix}}\ln(\tau_{\text{mix}})$:* [4]

$$\mathbb{E}\|\nabla f(\hat{\mathbf{x}}_T)\|^2 = \tilde{\mathcal{O}}\left(\frac{\Delta L\tau_{\text{mix}}}{T} + \frac{\sqrt{L\Delta\bar{\sigma}^2\tau_{\text{mix}}} + \bar{\sigma}^2}{\sqrt{T}}\right)\,,$$

   where $\hat{\mathbf{x}}_T$ is drawn uniformly at random amongst $\mathbf{x}_0,\dots,\mathbf{x}_{T-1}$.

2. *If f additionally verifies a μ-PL inequality (for any $\mathbf{x}\in\mathbb{R}^d$, $\|\nabla f(\mathbf{x})\|^2) \geq 2\mu(f(\mathbf{x}) - f(\mathbf{x}^\star))$), for a constant time-horizon dependent step size γ, the iterates generated by*

[3] this assumption could be replaced by a more relaxed noise assumption of the form $\|\nabla f_v(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq M\|\nabla f(\mathbf{x})\|^2 + \sigma_v^2$

[4] $\tilde{\mathcal{O}}$ hides logarithmic factors

Equation (7.2) satisfy, for $T \geq 2\tau_{\text{mix}} \ln(\tau_{\text{mix}})$ *a numerical constant* $c > 0$*, and* $\kappa = L/\mu$*, with* $F_T = \mathbb{E}\left[f(\mathbf{x}_T) - f(\mathbf{x}^\star)\right]$*:*

$$F_T \leq e^{-\frac{cT}{\kappa \tau_{\text{mix}} \ln(T)}} \Delta + \tilde{\mathcal{O}}\left(\frac{\tau_{\text{mix}} \bar{\sigma}^2}{\mu T}\right),$$

The proof of Theorem 7.3 is quite tedious, and is deferred to Appendix 7.B, where we enforce a delay of order τ_{mix} and rely on recent analyses of delayed SGD and SGD with biased gradients. As explained in the introduction, removing the bounded gradient assumption present in previous works [JRJ10, SSY18, DAJJ11] that study Markov chain SGD (in the mirror setting, or with subdifferentials), and replacing it by a much milder and classical assumption of bounded gradient dissimilarities [KKM+20], we thus still managed to obtain similar rates. Further, if f verifies a μ-PL inequality (if for any $\mathbf{x} \in \mathbb{R}^d$, $\|\nabla f(\mathbf{x})\|^2) \geq 2\mu(f(\mathbf{x}) - f(\mathbf{x}^\star))$), we have an almost-linear rate of convergence: this is the first rate under μ-PL or strong convexity assumptions for MC-SGD-like algorithms, that we even refine further in next subsection.

7.4.2. Tight rates and linear convergence in the interpolation regime

We now study MC-SGD under the following assumptions, to derive faster rates, that only depend on the sampling noise at the optimum. The interpolation regime – often related to overparameterization – refers to the case where there exists a model $\mathbf{x}^\star \in \mathbb{R}^d$ minimizing all f_v for $v \in \mathcal{V}$, leading to $\sigma_\star^2 = 0$ in Assumption 7.4.4, and to a linear convergence rate below.

Assumption 7.4.3. *Functions* f_v *are* L*-smooth and* μ*-strongly convex. We denote* $\kappa = L/\mu$*.*

Assumption 7.4.4 (Noise at the optimum)**.** *Let* $\mathbf{x}^\star$ *be a minimizer of* f*. We assume that for some* $\sigma_\star \geq 0$*, we have for all* $v \in \mathcal{V}$*:*

$$\|\nabla f_v(\mathbf{x}^\star)\|^2 \leq \sigma_\star^2 .$$

Theorem 7.4 (Unified analysis)**.** *Under Assumptions 7.4.3 and 7.4.4, the sequence generated by (7.2) satisfies, if* $\gamma L < 1$*:*

$$\mathbb{E}\left[\|\mathbf{x}_T - \mathbf{x}^\star\|^2\right] \leq 2(1-\gamma\mu)^T \|\mathbf{x}_0 - \mathbf{x}^\star\|^2 + 2\frac{\gamma^3 T L}{\mu} \sum_{0 \leq s \leq T} (1-\gamma\mu)^{T-s} \mathbb{E}\left[\left\|\sum_{s \leq t < T} \nabla f_{v_t}(\mathbf{x}^\star)\right\|^2\right].$$

In the interpolation regime, $\nabla f_v(\mathbf{x}^\star) = 0$ *for all* $v \in \mathcal{V}$*, so that:*

$$\mathbb{E}\left[\|\mathbf{x}_T - \mathbf{x}^\star\|^2\right] \leq 2(1-\gamma\mu)^T \|\mathbf{x}_0 - \mathbf{x}^\star\|^2 .$$

Proof of Theorem 7.4. Fix some $y_0 \in \mathbb{R}^d$ and let $(\mathbf{y}_t)$ be defined with the recursion

$$\mathbf{y}_{t+1} = \mathbf{y}_t - \gamma \nabla f_{v_t}(\mathbf{x}^\star, \xi_t) .$$

We now introduce and prove the following lemma.

Lemma 7.4.1. *For any* $\mathbf{y}_t \in \mathbb{R}^d$ *and* $t \geq 0$*, denoting* $\mathbf{y}_{t+1} = \mathbf{y}_t - \gamma \nabla f_{v_t}(\mathbf{x}^\star)$*, we have:*

$$\|\mathbf{x}_{t+1} - \mathbf{y}_{t+1}\|^2 \leq (1-\gamma\mu)\|\mathbf{x}_t - \mathbf{y}_t\|^2 + \gamma L \|\mathbf{y}_t - \mathbf{x}^\star\|^2 .$$

Proof of Lemma 7.4.1. Denote $f_t = f_{v_t}(\cdot)$. We expand:

$$\|\mathbf{x}_{t+1} - \mathbf{y}_{t+1}\|^2 = \|\mathbf{x}_t - \mathbf{y}_t\|^2 - 2\gamma\langle \nabla f_t(\mathbf{x}_t) - \nabla f_t(\mathbf{x}^\star), \mathbf{x}_t - \mathbf{y}_t\rangle + \gamma^2 \|\nabla f_t(\mathbf{x}_t) - \nabla f_t(\mathbf{x}^\star)\|^2 .$$

Using the *three points equality* as [MKR20], we have:

$$-2\gamma\langle\nabla f_t(\mathbf{x}_t) - \nabla f_t(\mathbf{x}^\star), \mathbf{x}_t - \mathbf{y}_t\rangle = -2\gamma D_{f_t}(\mathbf{y}_t, \mathbf{x}_t) - 2\gamma D_{f_t}(\mathbf{x}_t, \mathbf{x}^\star) + 2\gamma D_{f_t}(\mathbf{y}_t, \mathbf{x}^\star)\,.$$

First, $-2\gamma D_{f_t}(\mathbf{y}_t, \mathbf{x}_t) \leq -\gamma\mu\|\mathbf{x}_t - \mathbf{y}_t\|^2$ using strong convexity. Then, $-2\gamma D_{f_t}(\mathbf{x}_t, \mathbf{x}^\star)$ cancels the term $\gamma^2\|\nabla f_t(\mathbf{x}_t) - \nabla f_t(\mathbf{x}^\star)\|^2 \leq 2\gamma^2 L D_{f_t}(\mathbf{x}_t, \mathbf{x}^\star)$ for $\gamma \leq 1/L$, using smoothness of f_t. And finally, using smoothness again, $2\gamma D_{f_t}(\mathbf{y}_t, \mathbf{x}^\star) \leq \gamma L\|\mathbf{y}_t - \mathbf{x}^\star\|^2$, concluding the proof. □

Unrolling Lemma 7.4.1, we have, for a fixed time horizon T:

$$\|\mathbf{x}_T - \mathbf{y}_T\|^2 \leq (1-\gamma\mu)^T\|\mathbf{x}_0 - \mathbf{y}_0\|^2 + \gamma L\sum_{t<T}(1-\gamma\mu)^{T-t}\|\mathbf{y}_t - \mathbf{x}^\star\|^2\,.$$

This is possible, since the descent lemma is deterministic, in the sense that no expectations are taken so far. Since we want control over the distance to the optimum, we wish to have $\mathbf{y}_T = \mathbf{x}^\star$, leading to:

$$\mathbf{y}_0 = \mathbf{x}^\star + \gamma\sum_{t<T}\nabla f_{v_t}(\mathbf{x}^\star, \xi_t)\,, \quad \mathbf{y}_s = \mathbf{x}^\star + \gamma\sum_{s\leq t<T}\nabla f_{v_t}(\mathbf{x}^\star, \xi_t)\,, \quad s < T\,.$$

We thus have:

$$\mathbb{E}\|\mathbf{x}_T - \mathbf{x}^\star\|^2 \leq 2(1-\gamma\mu)^T\left(\mathbb{E}\|\mathbf{x}_0 - \mathbf{x}^\star\|^2 + \gamma^2\mathbb{E}\left[\left\|\sum_{s<T}\nabla f_{v_s}(\mathbf{x}^\star)\right\|^2\right]\right)$$
$$+ \gamma^3 L\sum_{t<T}(1-\gamma\mu)^{T-t}\mathbb{E}\left[\left\|\sum_{t\leq s<T}\nabla f_{v_s}(\mathbf{x}^\star)\right\|^2\right].$$

□

The result in Theorem 7.4 is in fact true irrespectively of the sequence (v_t) chosen: it does not require (v_t) to specifically be a Markov chain. This property is used in the next Corollary, that also highlights the fact that by studying distance to the optimum, a condition number is lost in the process. This is the case in many previous analyses of other different algorithms (*e.g.*, Bregman/Mirror-SGD [DEH21] or SGD with random-resfhuffling [MKR20], which is in fact a particular instance of MC-SGD, that our analysis recovers), that study distances to the optimum (with respect to some mirror map, in the case of Mirror SGD), and therefore obtain an extra κ factor in the noise term. Theorem 7.4 is proved by generalizing the proof technique of [MKR20] to arbitrary orderings and for unbounded time horizons.

Remark 7.4.1 (Random resfhuffling)**.** *A special case of Theorem 7.4 is SGD with random reshuffling. By analyzing SGD with random-reshuffling as SGD with a Markovian ordering (on an extended state space), Theorem 1,2 also recover rates for SGD with random reshuffling for which we have $\tau_{\text{mix}} = n$. Moreover, since Theorem 7.4 generalizes Theorem 1 of [MKR20], we also recover their rate as a special case by bounding each term $\mathbb{E}\left[\left\|\sum_{s\leq t<T}\nabla f_{v_t}(\mathbf{x}^\star)\right\|^2\right]$.*

We specify Theorem 7.4 under a Markovian sampling scheme in next corollary: the noise term at the optimum takes the form τ_{mix}/T.

Corollary 7.4.1 (MC-SGD, interpolation)**.** *Under Assumptions 7.4.3 and 7.4.4, for $T \geq 1$*

and for a well chosen stepsize $\gamma > 0$, the iterates generated by (7.2) satisfy:

$$\mathbb{E}\left[\|\mathbf{x}_T - \mathbf{x}^\star\|^2\right] \leq 2e^{-\frac{T}{\kappa}}\|\mathbf{x}_0 - \mathbf{x}^\star\|^2 + \tilde{\mathcal{O}}\left(\frac{L\tau_{\mathrm{mix}}\left(\frac{1}{4}\right)\sigma_\star^2}{\mu^3 T}\right).$$

Proof of Corollary 7.4.1. This corollary is a consequence of Theorem 7.4 and the following lemma.

Lemma 7.4.2. *For any $T \geq 1$, we have:*

$$\mathbb{E}\left[\left\|\sum_{t<T}\nabla f_{v_t}(\mathbf{x}^\star)\right\|^2\right] \leq T\sigma_\star^2 + \sigma_\star^2\sum_{t<T}\mathrm{d_{TV}}(P^t_{v_0,\cdot},\pi^\star) + 2\sigma_\star^2\sum_{s<t<T}\mathrm{d_{TV}}(t-s)\,,$$

where $\mathrm{d_{TV}}(r) = \sup\left\{\mathrm{d_{TV}}\left((P^r)_{v,\cdot},\pi^\star\right), v\in\mathcal{V}\right\}$ for $r\in\mathbb{N}$, so that:

$$\mathbb{E}\left[\left\|\sum_{t<T}\nabla f_{v_t}(\mathbf{x}^\star)\right\|^2\right] \leq \sigma_\star^2\big(4\tau_{\mathrm{mix}}(1/4) + T(1+8\tau_{\mathrm{mix}}(1/4))\big)\,.$$

Proof. We have:

$$\begin{aligned}\mathbb{E}\left[\left\|\sum_{t<T}\nabla f_{v_t}(\mathbf{x}^\star)\right\|^2\right] &= \mathbb{E}\left[\left(\sum_{t<T}\nabla f_{v_t}(\mathbf{x}^\star)\right)^\top\left(\sum_{t<T}\nabla f_{v_t}(\mathbf{x}^\star)\right)\right]\\ &= \sum_{t<T}\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}^\star)\|^2\right] + 2\sum_{s<t<T}\mathbb{E}\left[\langle\nabla f_{v_s}(\mathbf{x}^\star),\nabla f_{v_t}(\mathbf{x}^\star)\rangle\right].\end{aligned}$$

Denote $\mathbf{G}_\star = (\nabla f_v(\mathbf{x}^\star))_{v\in\mathcal{V}} \in \mathbb{R}^{\mathcal{V}\times d}$. For the first term above,

$$\begin{aligned}\sum_{t<T}\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}^\star)\|^2\right] &= \sum_{t<T}\mathbb{E}\left[\|\mathbf{G}_{\star,v_s}\|^2\right]\\ &= \sum_{t<T}\left(\sigma_\star^2 + \sum_{v\in\mathcal{V}}\left(\mathbb{P}\left(v_t = v\right) - \pi_v\right)\|\mathbf{G}_{\star,v}\|^2\right)\\ &\leq T\sigma_\star^2 + \sigma_\star^2\sum_{t<T}\mathrm{d_{TV}}(P^t_{v_0,\cdot},\pi^\star)\,.\end{aligned}$$

Finally,

$$\begin{aligned}\sum_{s<t<T}\mathbb{E}\left[\langle\nabla f_{v_s}(\mathbf{x}^\star),\nabla f_{v_t}(\mathbf{x}^\star)\rangle\right] &= \sum_{s<t<T}\mathbb{E}\left[\langle\mathbf{G}_{\star,v_s},\mathbf{G}_{\star,v_t}\rangle\right]\\ &= \sum_{s<t<T}\sum_{v,w\in\mathcal{V}}(P^s)_{v_0,v}(P^{t-s})_{v,w}\mathbf{G}_{\star,v}^\top\mathbf{G}_{\star,w}\\ &= \sum_{s<t<T}\sum_{v,w\in\mathcal{V}}(P^s)_{v_0,v}\big((P^{t-s})_{v,w} - \frac{1}{n}\big)\mathbf{G}_{\star,v}^\top\mathbf{G}_{\star,w}\\ &\leq \sum_{s<t<T}\sum_{v,w\in\mathcal{V}}(P^s)_{v_0,v}\big|(P^{t-s})_{v,w} - \frac{1}{n}\big|\sigma_\star^2\\ &= \sigma_\star^2\sum_{s<t<T}\sum_{v\in\mathcal{V}}(P^s)_{v_0,v}\sum_{w\in\mathcal{V}}\big|(P^{t-s})_{v,w} - \frac{1}{n}\big|\end{aligned}$$

$$\leq \sigma_\star^2 \sum_{s<t<T} \mathrm{d}_{\mathrm{TV}}(t-s)\,,$$

where $\mathrm{d}_{\mathrm{TV}}(r) = \sup\{\mathrm{d}_{\mathrm{TV}}((P^r)_{v,\cdot}, \pi^\star)\,, v \in \mathcal{V}\}$ for $r \in \mathbb{N}$.

We finally bound $\sum_{t<T} \mathrm{d}_{\mathrm{TV}}(t)$. Using Chapter 4.5 of [LPW06], we have $\tau_{\mathrm{mix}}(\varepsilon) \leq (\log_2(\varepsilon^{-1}) + 1)\tau_{\mathrm{mix}}(1/4)$, so that for any $t \geq 0$, $d_{\mathrm{TV}}(t) \leq 2^{-t/\tau_{\mathrm{mix}}(1/4)+1}$. Hence,

$$\sum_{t<T} \mathrm{d}_{\mathrm{TV}}(t) \leq \frac{2}{1 - 2^{-1/\tau_{\mathrm{mix}}(1/4)}} \leq 4\tau_{\mathrm{mix}}(1/4)\,,$$

and

$$\sum_{s<t<T} \mathrm{d}_{\mathrm{TV}}(t-s) \leq 4T\tau_{\mathrm{mix}}(1/4)\,,$$

concluding the proof. □

□

This result is stronger than Theorem 7.3.2, for *(i)* noise amplitude and gradient dissimilarities only need to be bounded at the optimum; *(ii)* the "optimization term" (the first one) is not slowed down by the mixing time. This comes at the cost of strong convexity assumptions, stronger than a μ-PL inequality for f. The term $\frac{\tau_{\mathrm{mix}}\sigma_\star}{T}$ cannot be removed in the general case, as next proposition shows. Hence, since the two other terms have optimal dependency in terms of Markov-chain and noise related quantities, our analysis ends up being sharp.

Corollary 7.4 together with the following proposition are an extension of [NWB+20], who proved similar results for MC-SGD with constant stepsize on least square problems on Markovian data of a certain form (for linear online system identification).

Proposition 7.4.1. *For any $\mathcal{V}$ (such that $|\mathcal{V}| \geq 2$) and $\tau > 1$, there exists a Markov chain on $\mathcal{V}$ of relaxation time τ, functions $(f_v)_{v\in\mathcal{V}}$ and $\mathbf{x}_0 \in \mathbb{R}^d$ such that given any stepsize γ, the iterates of Equation (7.2) output $\mathbf{x}_T$ for any $T > 0$ verifying $\|\mathbf{x}_T - \mathbf{x}_0\|^2 = \tilde{\Omega}(\tau\sigma_\star^2/T)$, and the assumptions of Theorem 7.4 hold.*

7.4.3. MC-SGD with local noise

In the two previous subsections, we analyzed SGD with Markovian sampling schemes, where the stochasticity only came from the Markov chain $(v_k)_{k\geq 0}$. We now generalize the analysis and results to SGD with both Markovian sampling, and local noise, by studying the sequence:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma_t \mathbf{g}_t\,. \tag{7.4}$$

We now formulate the form stochastic gradients $\mathbf{g}_t$ can take.

Assumption 7.4.5. *For all $v \in \mathcal{V}$, the function f_v satisfies $f_v(\mathbf{x}) = \mathbb{E}\left[F_v(\mathbf{x}, \xi_v)\right]$ for all $\mathbf{x} \in \mathbb{R}^d$, where $\xi_v \sim \mathcal{D}_v$. Furthermore, there exists a Markov-chain $(v_t)_{t\geq 0}$ such that for all $t \geq 0$,*

$$\mathbf{g}_t = \nabla_{\mathbf{x}} F_{v_t}(\mathbf{x}_t, \xi_t)\,,$$

where $\xi_t \sim \mathcal{D}_{v_t}|v_t$ is independent from $v_0, \ldots, v_{t-1}$ and $\xi_0, \ldots, \xi_{t-1}$.

A direct consequence of Assumption 7.4.3 is that $\mathbb{E}\left[\mathbf{g}_t|\mathbf{x}_t, v_t\right] = \nabla f_{v_t}(\mathbf{x}_t)$. Two main applications of Assumption 7.4.3 are:

1. **Local sampling.** If $f_v(\mathbf{x}) = \frac{1}{m}\sum_{i=1}^m f_{v,i}(\mathbf{x})$ (agent v has m local samples), agent m may use only a batch $\mathcal{B} \subset [m]$ of its samples, leading to stochastic gradients $\mathbf{g}_t$ in (7.4) of the form:
$$\mathbf{g}_t = \frac{1}{|\mathcal{B}_t|}\sum_{i\in\mathcal{B}_t} \nabla f_{v_t,i}(\mathbf{x}_t)\,,$$

for random batches $(\mathcal{B}_t)_{t\geq 0}$.

2. **Differential privacy.** Adding local noise (*e.g.*, additive Gaussian random noise) enforces differential privacy under suitable assumptions. A private decentralized token algorithm is then **Differentially Private MC-SGD** (DP-MC-SGD), with iterates (7.4) where $\mathbf{g}_t$ satisfies
$$\mathbf{g}_t = \nabla f_{v_t}(\mathbf{x}_t) + \eta_t\,, \tag{7.5}$$
where (v_t) is the Markov chain (random walk performed by the token on the communication graph), and $\eta_t \sim \mathcal{N}(0, \sigma_t^2 I_d)$ is sampled independently from the past, to enforce differential privacy.

Under Assumption 7.4.3, a direct generalization of Theorem 7.4 and Corollary 7.4.1 is the following.

Theorem 7.5 (MC-SGD with local noise)**.** *Assume that Assumptions 7.4.5,7.4.4 holds, each $F_v(\cdot,\xi)$ is μ-strongly convex L-smooth, and there exists $\sigma^2_{\rm local}$ such that:*
$$\mathbb{E}\left[\|\mathbf{g}_t - \nabla f_{v_t}(\mathbf{x}_t)\|^2 | \mathbf{x}_t, v_t\right] \leq \sigma^2_{\rm local}\,.$$
Then, for a well chosen $\gamma > 0$, the iterates generated by (7.4) satisfy:
$$\mathbb{E}\left[\|\mathbf{x}_T - \mathbf{x}^\star\|^2\right] \leq 2e^{-\frac{T}{\kappa}}\|\mathbf{x}_0 - \mathbf{x}^\star\|^2 + \tilde{\mathcal{O}}\left(\frac{L}{\mu^3 T}\big(\sigma^2_{\rm local} + \tau_{\rm mix}\big(\frac{1}{4}\big)\sigma^2_\star\big)\right)\,.$$

Importantly, and as one would have expected, local noise is **not** impacted by the mixing time of the underlying random walk. While we did not pursue in this direction, this observation could easily be made under other regularity assumptions, and such a result would hold for instance under the assumptions of Theorem 1 or 2. While Differentially Private MC-SGD sounds appealing for performing decentralized and differentially private optimization, we here only provided a utility analysis, the privacy analysis being left for future works.

7.5. Analysis of Markov-Chain SAG

Algorithm 7.1: Markov Chain SAG (MC-SAG)

Input: $\mathbf{x}_0 \in \mathbb{R}^d$, $\mathbf{h}_v \in \mathbb{R}^d$ for $v \in \mathcal{V}$ and $\bar{\mathbf{h}}_0 \in \mathbb{R}^d$, stepsizes $\gamma_t > 0$, $v_0 \in \mathcal{V}$
for $t = 0, 1, \ldots$ **do**
  Compute $\nabla f_{v_t}(\mathbf{x}_t)$
  $\bar{\mathbf{h}}_{t+1} = \bar{\mathbf{h}}_t + \frac{1}{n}\big(\nabla f_{v_t}(\mathbf{x}_t) - \mathbf{h}_{v_t}\big)$
  $\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma_t \bar{\mathbf{h}}_{t+1}$
  $\mathbf{h}_{v_t} \longleftarrow \nabla f_{v_t}(\mathbf{x}_t)$
  Sample $v_{t+1} \sim P_{v_t,\cdot}$.
end for

After providing convergence guarantees for the most natural algorithm (MC-SGD) under a Markov chain sampling on the set $\mathcal{V}$, we prove that one can achieve a rate of order $1/T$ (rather than the $1/\sqrt{T}$ previously obtained) in the smooth setting, by applying the variance reduction techniques present in [SLRB17], that first introduced the Stochastic Averaged Gradient algorithm, together with a time-adaptive stepsize policy described below. Our faster rate with variance reduction leads of a dependency on $\tau_{\rm hit}$ instead of $\tau_{\rm mix}$; since we do not

make any other assumption other than smoothness, this is unavoidable in light of our lower bound (Theorem 7.2).

MC-SAG The MC-SAG algorithm is described in Algorithm 7.1. The recursion leading to the iterate $\mathbf{x}_t$ can then be summarized as, for stepsizes $(\gamma_t)_{t\geq 0}$, under the initialization $\mathbf{h}_v = \nabla f_v(\mathbf{x}_0)$ and $\bar{\mathbf{h}} = \nabla f(\mathbf{x}_0)$:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{\gamma_t}{n} \sum_{v \in \mathcal{V}} \nabla f_v(\mathbf{x}_{d_v(t)}) \,, \tag{7.6}$$

where for $v \in \mathcal{V}$, we define $d_v(t) = \sup\{s \leq t \,|\, v_s = v\}$ as the last previous iterate at which v was the current state of the Markov chain. By convention, if the set over which the supremum is taken is empty, we set $d_v(t) = 0$. We handle both the initialization described just above for $\mathbf{h}_v, \bar{\mathbf{h}}$ and arbitrary initialization in our analysis below.

In the same way that MC-SGD reduces to vanilla SGD if (v_t) is an *i.i.d.* uniform sampling over $\mathcal{V}$, MC-SAG boils down to the SAG algorithm [SLRB17] in that case and under the initialization $\mathbf{h}_v = \nabla f_v(\mathbf{x}_0)$ and $\bar{\mathbf{h}} = \nabla f(\mathbf{x}_0)$. In a decentralized setting, nodes keep in mind their last gradient computed (variable $\mathbf{h}_v$ at node v). At all times, $\bar{\mathbf{h}}_t$ is an average of these $\mathbf{h}_v$ over the graph, and is, in the same way as $\mathbf{x}_t$, updated along the random walk. The MC-SAG algorithm is thus perfectly adapted to decentralized optimization.

Time-adaptive stepsize policy To obtain our convergence guarantees, a time-adaptive stepsize policy (γ_t) is used, as in Asynchronous SGD [MBEW22] to obtain delay-independent guarantees. For $t \geq 0$, let the stepsize γ_t be defined as:

$$\gamma_t = \frac{1}{2L\big(\tau_{\text{hit}} + \max_{v\in\mathcal{V}}(t - d_v(t))\big)} \,. \tag{7.7}$$

Denoting $\tau_t = \max_{v\in\mathcal{V}}(t - d_v(t))$, this quantity can be tracked down during the optimization process. Indeed, if agent v_t receives τ_{t-1} together with $(\mathbf{x}_t, \bar{\mathbf{h}}_t)$, she may compute τ_t as:

$$\tau_t = \max\left(\tau_{t-1} + 1 \,,\, t - d_{v_t}(t)\right) ,$$

where $t - d_{v_t}(t)$ is the number of iterations that took place since the last time the Markov chain state was v_t. Hence, if agents keep track of the number of iterations, the adaptive stepsize policy (7.7) can be used in Algorithm 7.1, as long as agent v_t sends (τ_t, t) to v_{t+1}, yielding the following result.

We now present the convergence results for MC-SAG. (v_t) is in this section assumed to be a Markov chain on $\mathcal{V}$ of finite hitting time τ_{hit}. Importantly, the next Theorem does not require any additional assumption on (v_t) such as reversibility, or even that it has a stationary probability that is the uniform distribution: the non-symmetric but easily implementable transition probabilities $P_{v,w} = 1/(d_v + 1)$ for $w = v$ or $w \sim v$ can be used here, as well as non-reversible random walks than can have much smaller mixing and hitting times. The function f studied is here independent of the Markov chain, and is defined as in (7.3), the uniformly averaged function over all states $v \in \mathcal{V}$ (or over all agents in the network).

Theorem 7.6 (MC-SAG)**.** *Assume that Assumption 7.4.2 holds and that the Markov chain has a finite hitting time (for an arbitrary invariant probability).*

1. *Under the initialization:*
$$\mathbf{h}_v = \nabla f_v(\mathbf{x}_0) \,,$$
$$\bar{\mathbf{h}} = \nabla f(\mathbf{x}_0) \,,$$
using the adaptive stepsize policy defined in Equation (7.7), the sequence generated by

Algorithm 7.1 satisfies, for any $T > 0$:

$$\mathbb{E}\left[\min_{t<T} \|\nabla f(\mathbf{x}_t)\|^2\right] \leq 8L\big(f(\mathbf{x}_0) - f(\mathbf{x}^\star)\big)\frac{\tau_{\text{hit}}\ln(n)}{T}\,.$$

2. *Under any arbitrary initialization that satisfies $\bar{\mathbf{h}} = \frac{1}{n}\sum_{v\in\mathcal{V}} \mathbf{h}_v$, using the adaptive step-size policy defined in Equation (7.7), the sequence generated by Algorithm 7.1 satisfies, for any $T > 0$:*

$$\mathbb{E}\left[\min_{t<T} \|\nabla f(\mathbf{x}_t)\|^2\right] \leq 16L\Delta\frac{\tau_{\text{hit}}\ln(n)}{T}\,,$$

where

$$\Delta = f(\mathbf{x}_0) - f(\mathbf{x}^\star) + \frac{1}{8n}\sum_{v\in\mathcal{V}} \|\nabla f_v(\mathbf{x}_0) - \mathbf{h}_v\|^2\,.$$

Theorem 7.6 is proved in Appendix 7.6. Up to the logarithmic factor in n, the rates in Theorem 7.6 are the non-accelerated versions of the lower-bound in Theorem 7.2.

Proof of Theorem 7.6.1. We begin classically by proving a descent lemma. This lemma is deterministic, in the sense that not means $\mathbb{E}$ are present, and it therefore does not use the Markovian properties of the Markov chain. MC-SAG uses biased gradients, even in the case where v_t are *i.i.d.*, since the algorithm SAG [SLRB17] is inherently biased (making it unbiased leads to the SAGA iterations [DBLJ14]).

Let $\mathbf{G}_t = \bar{\mathbf{h}}_{t+1}$ for $t \geq 0$, so that $\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma_t \mathbf{G}_t$. We recall that for $v \in \mathcal{V}$ and $t \geq 0$, $p_v(t)$ is the next time (strictly) the chain hits node v, while $d_v(t)$ is either the last time the chain was at the state v (if that happened), or 0 in v has not yet been visited.

Lemma 7.5.1. *Assume that f is L-smooth. Then, for any $t \geq 0$, we have:*

$$f(\mathbf{x}_{t+1}) - f(\mathbf{x}_t) \leq -\frac{\gamma_t}{2}\|\nabla f(\mathbf{x}_t)\|^2 - \frac{\gamma_t}{4}\|\mathbf{G}_t\|^2 + \frac{\gamma_t L^2}{2n}\sum_{v\in\mathcal{V}}\left\|\sum_{s=d_v(t)}^{t-1}\gamma_s\mathbf{G}_s\right\|^2\,.$$

Proof of Lemma 7.5.1. For $t \geq 0$ and $v \in \mathcal{V}$, let $p_v(t) = \inf\{s > t | v_s = v\}$ and $d_v(t) = \sup\{s \leq t | v_s = v\}$ be the next and the last previous iterates for which $v_t = v$ ($d_v(t) = 0$ by convention, if v has not yet been visited). Denote $F_t = \mathbb{E}\left[f(\mathbf{x}_t) - f(\mathbf{x}^\star)\right]$. We have, using smoothness:

$$f(\mathbf{x}_{t+1}) \leq f(\mathbf{x}_t) - \gamma_t\langle\nabla f(\mathbf{x}_t), \mathbf{G}_t\rangle + \frac{\gamma_t^2 L}{2}\|\mathbf{G}_t\|^2$$

Together with $\langle\nabla f(\mathbf{x}_t), \mathbf{G}_t\rangle = \frac{1}{2}(\|\nabla f(\mathbf{x}_t)\|^2 + \|\mathbf{G}_t\|^2 - \|\nabla f(\mathbf{x}_t) - \mathbf{G}_t\|^2)$, we obtain:

$$\begin{aligned} f(\mathbf{x}_{t+1}) &\leq f(\mathbf{x}_t) - \frac{\gamma_t}{2}\|\nabla f(\mathbf{x}_t)\|^2 + \|\mathbf{G}_t\|^2 - \|\nabla f(\mathbf{x}_t) - \mathbf{G}_t\|^2 + \frac{\gamma_t^2 L}{2}\|\mathbf{G}_t\|^2 \\ &\leq f(\mathbf{x}_t) - \frac{\gamma_t}{2}\|\nabla f(\mathbf{x}_t)\|^2 - \frac{\gamma_t}{4}\|\mathbf{G}_t\|^2 + \frac{\gamma_t}{2}\|\nabla f(\mathbf{x}_t) - \mathbf{G}_t\|^2\,, \end{aligned}$$

as long as $\gamma_t \leq 1/(2L)$. We thus need to upperbound the bias $\|\nabla f(\mathbf{x}_t) - \mathbf{G}_t\|^2$. We have:

$$\begin{aligned} \|\nabla f(\mathbf{x}_t) - \mathbf{G}_t\|^2 &= \left\|\frac{1}{n}\sum_{v\in\mathcal{V}} \nabla f_v(\mathbf{x}_{d_v(t)}) - \nabla f_v(\mathbf{x}_t)\right\|^2 \\ &\leq \frac{1}{n}\sum_{v\in\mathcal{V}} \left\|\nabla f_v(\mathbf{x}_{d_v(t)}) - \nabla f_v(\mathbf{x}_t)\right\|^2\,. \end{aligned}$$

Fix some v in $\mathcal{V}$. We have $\left\|\nabla f_v(\mathbf{x}_t) - \nabla f_v(\mathbf{x}_{d_v(t)})\right\|^2 \leq L^2 \left\|\sum_{s=d_v(t)}^{t-1} \gamma_s \mathbf{G}_s\right\|^2$, leading to:

$$f(\mathbf{x}_{t+1}) - f(\mathbf{x}_t) \leq -\frac{\gamma_t}{2}\|\nabla f(\mathbf{x}_t)\|^2 - \frac{\gamma_t}{4}\|\mathbf{G}_t\|^2 + \frac{\gamma_t L^2}{2n}\sum_{v\in\mathcal{V}}\left\|\sum_{s=d_v(t)}^{t-1}\gamma_s\mathbf{G}_s\right\|^2.$$

□

We begin with

$$f(\mathbf{x}_{t+1}) - f(\mathbf{x}_t) \leq -\frac{\gamma_t}{2}\|\nabla f(\mathbf{x}_t)\|^2 - \frac{\gamma_t}{4}\|\mathbf{G}_t\|^2 + \frac{\gamma_t L^2}{2n}\sum_{v\in\mathcal{V}}\left\|\sum_{s=d_v(t)}^{t-1}\gamma_s\mathbf{G}_s\right\|^2,$$

as a starting point. For $v \in \mathcal{V}$,

$$\begin{aligned}\gamma_t\left\|\sum_{s=d_v(t)}^{t-1} L^2\gamma_s\mathbf{G}_s\right\|^2 &\leq \sum_{s=d_v(t)}^{t-1}(t-d_v(t))L^2\gamma_t\gamma_s^2\|\mathbf{G}_s\|^2 \\ &\leq \sum_{s=d_v(t)}^{t-1} L\gamma_s^2\|\mathbf{G}_s\|^2,\end{aligned}$$

since $\gamma_t \leq 1/(L(t-d_v(t)))$. Summing for $t < T$, we obtain:

$$\sum_{t<T}\frac{\gamma_t}{2}\|\nabla f(\mathbf{x}_t)\|^2 \leq F_0 - \sum_{t<T}\frac{\gamma_t}{2}\|\mathbf{G}_t\|^2 + \sum_{t<T}\frac{1}{2n}\sum_{v\in\mathcal{V}}\sum_{s=d_v(t)}^{t-1} L\gamma_s^2\|\mathbf{G}_s\|^2.$$

Then,

$$\sum_{t<T}\sum_{s=d_v(t)}^{t-1} L\gamma_s^2\|\mathbf{G}_s\|^2 = \sum_{s<T}\|\mathbf{G}_s\|^2 L\gamma_s^2(p_v(s)-s).$$

For $\gamma_s \leq 1/(2L\tau_{\text{hit}})$, we have $\mathbb{E}\left[\|\mathbf{G}_s\|^2\gamma_s^2(p_v(s)-s)\right] \leq \frac{1}{2}\mathbb{E}\left[\|\mathbf{G}_s\|^2\gamma_s\right]$, so that:

$$\sum_{t<T}\frac{\gamma_t}{2}\|\nabla f(\mathbf{x}_t)\|^2 \leq F_0 + \sum_{t<T}\left(-\frac{\gamma_t}{2}\|\mathbf{G}_t\|^2 + K_t\right),$$

where $K_t = \frac{1}{4n}\sum_{v\in\mathcal{V}}\|\mathbf{G}_t\|^2 L\gamma_t^2(p_v(t)-t)$ verifies $\mathbb{E}[K_t|\mathcal{F}_t] \leq \|\mathbf{G}_t\|^2\gamma_t/4$ since $\gamma_t \leq 1/(2\tau_{\text{hit}})$, where $\mathcal{F}_t$ is the filtration up to time t:

$$\begin{aligned}\mathbb{E}[K_t|\mathcal{F}_t] &= \mathbb{E}\left[\frac{1}{4n}\sum_{v\in\mathcal{V}}\|\mathbf{G}_t\|^2 L\gamma_t^2(p_v(t)-t)|\mathcal{F}_t\right] \\ &= \frac{1}{4n}\sum_{v\in\mathcal{V}}\|\mathbf{G}_t\|^2 L\gamma_t^2\mathbb{E}[(p_v(t)-t)|\mathcal{F}_t] \quad \left(\mathbf{G}_t\,,\,\gamma_t \text{ are } \mathcal{F}_t\text{-measurable}\right) \\ &\leq \frac{1}{4n}\sum_{v\in\mathcal{V}}\|\mathbf{G}_t\|^2 L\gamma_t^2\tau_{\text{hit}} \quad \left(\text{since } \mathbb{E}[(p_v(t)-t)|\mathcal{F}_t] = \mathbb{E}[(p_v(t)-t)|v_t] \leq \tau_{\text{hit}}\right) \\ &\leq \frac{1}{8}\gamma_t\|\mathbf{G}_t\|^2 \quad \left(\text{since } \gamma_t \leq 1/(2\tau_{\text{hit}})\right).\end{aligned}$$

Finally,

$$\mathbb{E}\left[\min_{t<T}\|\nabla f(\mathbf{x}_t)\|^2\right] \leq \mathbb{E}\left[\frac{2F_0}{\sum_{t<T}\gamma_t}\right] + \mathbb{E}\left[\frac{\sum_{t<T}\left(-\frac{\gamma_t}{2}\|\mathbf{G}_t\|^2 + K_t\right)}{\sum_{t<T}\gamma_t}\right].$$

We now use the following Lemma 7.5.2.

Lemma 7.5.2. *Let $(a_t)_{t\geq 0}, (b_t)_{t\geq 0}$ be two sequences of real-valued random variables. Let $(\mathcal{F}_t)_{t\geq 0}$ be a filtration. Assume that b_t is positive and $\mathcal{F}_t$-measurable for all t, and that $\mathbb{E}[a_t|\mathcal{F}_t] \leq 0$. Then, denoting*

$$H_T = \frac{\sum_{t=0}^{T} a_t}{\sum_{t=0}^{T} b_t},$$

the sequence $(\mathbb{E}[H_T])_{T\geq 0}$ is non-increasing, so that $\mathbb{E}[H_T] \leq 0$ for all T.

Proof. For fixed $T \geq 1$, we have, using the fact tha b_t is $\mathcal{F}_T$ measurable for $t \leq T$

$$\begin{aligned}\mathbb{E}[H_T|\mathcal{F}_T] &= \frac{\mathbb{E}[a_T|\mathcal{F}_T] + \sum_{t<T}\mathbb{E}[a_t|\mathcal{F}_T]}{\sum_{t\leq T} b_t} \\ &\leq \frac{\sum_{t<T}\mathbb{E}[a_t|\mathcal{F}_T]}{\sum_{t\leq T} b_t} \\ &\leq \frac{\sum_{t<T}\mathbb{E}[a_t|\mathcal{F}_T]}{\sum_{t<T} b_t},\end{aligned}$$

using $\mathbb{E}[a_T|\mathcal{F}_T] \leq 0$ and $b_T > 0$. Consequently, taking the mean, we obtain $\mathbb{E}[H_T] \leq \mathbb{E}[H_{T-1}]$. □

Using both Lemma 7.5.2 and the above bound on $\mathbb{E}[K_t|\mathcal{F}_t]$, our last term is non-positive. Using Jensen inequality, we have:

$$\mathbb{E}\left[\min_{t<T}\|\nabla f(\mathbf{x}_t)\|^2\right] \leq \frac{2F_0}{T^2}\sum_{t<T}\mathbb{E}\left[\gamma_t^{-1}\right].$$

Since $\gamma_t^{-1} = 2L(\tau_{\text{hit}} + \sup_{v\in\mathcal{V}}(t - d_v(t)))$, we have $\sum_{t<T}\mathbb{E}\left[\gamma_t^{-1}\right] \leq 2LT(\tau_{\text{hit}} + \tau_{\text{cov}})$ using Lemma 7.5.3 below, concluding the proof.

Lemma 7.5.3. *For $t \geq 0$ and $v \in \mathcal{V}$, let $p_v(t) = \inf\{s > t | v_s = v\}$ and $d_v(t) = \sup\{s < t | v_s = v\}$ be the next and the last previous iterates for which $v_t = v$ ($d_v(t) = 0$ by convention, if v has not yet been visited). Assume that (v_t) has stationary distribution π. For $t \geq 0$, let $A_t = \sup_{v\in\mathcal{V}}\left(t - d_v(t)\right)$ and $B_t = \sup_{v\in\mathcal{V}}\left(p_v(t) - t\right)$. We have:*

$$\mathbb{E}[B_t|v_t = v] \leq \tau_{\text{cov}}, \quad \forall v \in \mathcal{V},$$

and for $T \geq 1$:

$$\sum_{t<T}\mathbb{E}[A_t] \leq T\tau_{\text{cov}}.$$

Proof. The first bound on B_t is obtained using the Markov property of the chain, and by definition of τ_{cov}. We have:

$$\mathbb{E}[A_t] = \sum_{k=0}^{t-1}\mathbb{P}(A_t \geq k).$$

For $t \geq 0$ fixed, we denote $d = \inf_v d_v(t)$, so that $A_t = t - d$. We then have the equality between the following events:

$$\{A_t \geq k\} = \{t - d \geq k\} = \{d \leq t - k\} = \{B_{t-k} \geq k\},$$

that all coincide with the event "there exists some $v \in \mathcal{V}$ such that for all $t - k \leq s \leq t$, $v_s \neq v$". Summing over $t < T$:

$$\begin{aligned}
\sum_{t<T} \mathbb{E}[A_t] &= \sum_{t<T} \sum_{k<t} \mathbb{P}(B_{t-k} \geq k) \\
&= \sum_{\ell<T} \sum_{s<T-\ell} \mathbb{P}(B_\ell \geq s) \\
&\leq \sum_{\ell<T} \sum_{s\geq 0} \mathbb{P}(B_\ell \geq s) \\
&\leq \sum_{\ell<T} \mathbb{E}[B_\ell] \\
&\leq T\tau_{\mathrm{cov}}\,.
\end{aligned}$$

□

□

Proof of Theorem 7.6.2. Starting from

$$\begin{aligned}
f(\mathbf{x}_{t+1}) - f(\mathbf{x}_t) \leq -\frac{\gamma_t}{2}\|\nabla f(\mathbf{x}_t)\|^2 - \frac{\gamma_t}{4}\|\mathbf{G}_t\|^2 \\
+ \frac{\gamma_t L^2}{n} \sum_{v\in\mathcal{V}} \left\| \sum_{s=d_v(t)}^{t-1} \gamma_s \mathbf{G}_s \right\|^2 + \frac{\gamma_t \mathbb{1}_{t<\tilde{\tau}_{\mathrm{cov}}}}{n} \sum_{v\in\mathcal{V}} \|\nabla f_v(\mathbf{x}_0) - \mathbf{h}_v\|^2,
\end{aligned}$$

and mimicking the proof with initialization, we obtain that for $\gamma_t^{-1} = 4L(\tau_{\mathrm{hit}} + \sup_{v\in\mathcal{V}}(t - d_v(t)))$, we have

$$\mathbb{E}\left[\min_{t<T} \|\nabla f(\mathbf{x}_t)\|^2\right] \leq \frac{2F_0}{T^2} \sum_{t<T} \mathbb{E}\left[\gamma_t^{-1}\right] + \frac{2\mathbb{E}[\tilde{\tau}_{\mathrm{cov}}]}{T} \frac{1}{n} \sum_{v\in\mathcal{V}} \|\nabla f_v(\mathbf{x}_0) - \mathbf{h}_v\|^2,$$

and we conclude the proof by noticing that $\mathbb{E}[\tilde{\tau}_{\mathrm{cov}}] \leq \tau_{\mathrm{cov}}$. □

7.6. Discussion of our results

7.6.1. Communication efficiency: comparison of our results with consensus-based approaches

We summarize the communication efficiencies in Table 7.1 (in terms of total number of communications required to reach an ε-stationary point), of classical gossip-based decentralized gradient methods (non-accelerated, since no accelerated method is known under our regularity assumptions). We consider the algorithm of [YJY19] (decentralized SGD with momemtum, state of the art decentralized gossip-based algorithm for this problem) with fixed communication matrix W on the graph G together with the Walkman algorithm [MYH+20] and our algorithms, for a Markov chain with transition matrix P. For the sake of comparison, we take as gossip matrix $W = P$.Consequently as shown in Table 7.1, *our algorithm (MC-SAG) always outperforms non-accelerated gossip-based decentralized gradient descent algorithms in terms of number of communications required to reach ε-stationary points.* Note that we do not claim the "overall superiority" of our approach over classical decentralized

optimization algorithms (the latter benefit from parallelization while ours do not), but a superiority in terms of communication efficiency.

Table 7.1 – *Number of communications required (# comm. below) to obtain an ε-stationary point $\mathbf{x}$ (verifying $\|\nabla f(\mathbf{x})\|^2 \leq \varepsilon$). Logarithmic/constant factors hidden.*

	A	B	Our work
# comm.	$\varepsilon^{-1}\lvert\mathcal{E}\rvert\tau_{\text{mix}}$	$\varepsilon^{-1}n\tau_{\text{mix}}^2$	$\varepsilon^{-2}\tau_{\text{mix}}$ [c] $\varepsilon^{-1}\tau_{\text{hit}}$ [d]

A: [YJY19].
B: [MYH+20].
[c] MC-SGD under Assumption 7.4.1.
[d] MC-SAG.

Table 7.2 – *Hitting and mixing times of some known graphs, for the simple random walk.*

	Cycle	d-dim. torus	Complete graph
τ_{hit}	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{1+\frac{1}{d}})$	$\mathcal{O}(n)$
$n\tau_{\text{mix}}$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^{1+\frac{2}{d}})$	$\mathcal{O}(n)$

The dependency on the quantity τ_{hit} we obtain (under no other assumptions than smoothness) is always better than the dependency on $n\tau_{\text{mix}}$ of previous works (using gossip communications or a random walker), since $\tau_{\text{hit}} \leq 2n\tau_{\text{mix}}$ always holds. As illustrated in Table 7.2 on some known graphs, this inequality is rather loose when the connectivity decreases (*i.e.* the mixing time increases), so that the speedup our results lead to is even more effective on ill-connected graphs; the difference between the two can scale up to a factor n. In fact, we proved in Section 7.2.2 that for d-regular and symetric graphs, we have:

$$\tau_{\text{hit}} \leq \frac{2|\mathcal{E}|\text{Diam}(G)}{d},$$

where Diam is the diameter of G. The dependency $n\tau_{\text{mix}}^2$ obtained in [MYH+20] (the only work that does not make bounded gradient assumptions) is prohibitive when graph connectivity decreases (n^3 on the grid, n^5 on the cycle). Our analysis does not rely on a reversibility assumption of the Markov chain, so that non symetric random walks can be used, therefore accelerating mixing; on the cycle for a non-symmetric random walk for instance, the hitting time decreases to $\mathcal{O}(n)$.

7.6.2. Using all gradient along the trajectory of (v_t) is provably more efficient

[SSY18] empirically motivated through empirical evidence the use of all gradients ∇f_{v_t} sampled along the trajectory of the Markov chain rather than waiting for the chain to mix before every stochastic gradient step in order to mimic the behavior of vanilla SGD. However, their rates (as well as those of [JRJ10, DAJJ11] and ours for MC-SGD) are functions of $S = T/\tau_{\text{mix}}$, and of order $1/S + 1/\sqrt{S}$. These are exactly what one would obtain, by waiting for τ_{mix} steps of the chain in order to have an approximate uniform sampling before each update! Consequently, there are no theoretical ground or evidence for using all the gradients along the trajectory of the Markov chain with these results, other than by doing so, one does not do worse than waiting for the chain to mix to mimic vanilla SGD. This is exactly the approach taken by [Hen22]: a gradient step is performed every τ_{mix} random walk steps. This is where MC-SAG and its guarantees that depend on T/τ_{hit} come in place. Under our assumptions, the rate of SAG for finding approximate stationary points when waiting for the

chain to mix before using a stochastic gradient is of order $n/S = n\tau_{\text{mix}}/T$ where $S = T/\tau_{\text{mix}}$ is the number of stochastic gradients used. We obtain τ_{hit}/T instead: hence, in cases where $\tau_{\text{hit}} = o(n\tau_{\text{mix}})$, using all stochastic gradients along the trajectory of the Markov chain - instead of waiting for mixing before performing a stochastic gradient step - provably helps. Hence, we here provided a realistic scenario where using all stochastic gradients proves to accelerate the rate; this was previously noticed in another setting with RER-SGD (SGD with reverse-experience replay, [KNJN21]).

7.6.3. Running-time complexity and robustness to "stragglers"

The total time it takes to run random walk-based decentralized algorithms depends on $T_{v\to w}$, the time it takes to compute a gradient at v, and then the communication time to send it to w. Using ergodicity of the Markov chain, the time $\text{time}^{\text{MC}}(T)$ it takes to run MC-SAG or MC-SGD for T iterations verifies:

$$\frac{\text{time}^{\text{MC}}(T)}{T} \to \sum_{(v,w)\in\mathcal{V}^2} \pi_v P_{v,w} T_{v\to w}\,,$$

where the limit is a weighted sum of the local computation/communication times, with weights summing to 1. Random-walk based decentralized algorithms are therefore robust to slow edges or nodes ("stragglers"), a property that synchronous gossip algorithms do not verify (their time complexity depends on $\max_{v,w} T_{v\to w}$), while studying asynchronous gossip is notoriously difficult (see Chapters 4 and 5).

Conclusion

Without variance reduction and under bounded data-heterogeneity assumptions, SGD under MC sampling is slowed down by a factor τ_{mix}, due to increased sampling variance. Using variance-reduction techniques, we obtain faster rates, that depend on τ_{hit} rather than $n\tau_{\text{mix}}$, which one would have expected by directly extending known results in the *i.i.d.* setting to our MC sampling schemes. Leveraging such a dependency yields a fast token algorithm (MC-SAG), robust to both ill-connectivity of the graph and data-heterogeneity.

Appendix of Chapter 7

7.A. Lower bound

We prove the smooth non-convex version of Theorem 7.2; the convex cases are proved in a similar way using exactly the same arguments, and the "most difficult function in the world", as defined by [Nes14], rather than the one used by [CDHS21], albeit the two are closely related.

Proof of Theorem 7.2. For $\mathbf{x} \in \ell_2$ and $k \in \mathbb{N}$, denote by $\mathbf{x}(k)$ its k^{th} coordinate. We split the function defined in Section 3.2 of [CDHS21] (inspired by the "most difficult function in the world" of [Nes14]) between two nodes $v, w \in \mathcal{V}$ maximizing $\mathbb{E}\left[\tau_w | v_0 = v\right]$, by setting $\pi_v f_v(\mathbf{x}) = \frac{1}{2}\sum_{k\geq 1} 2\mathbf{x}(2k)^2 - 2\mathbf{x}(2k-1)\mathbf{x}(2k) + \frac{1}{2}\alpha\mathbf{x}(0)^2 - b\mathbf{x}(0) + \frac{\alpha}{2}$ and $\pi_w f_w(\mathbf{x}) = \frac{1}{2}\sum_{k\geq 0} 2\mathbf{x}(2k+1)^2 - 2\mathbf{x}(2k+1)\mathbf{x}(2k)$ for some $b, \alpha > 0$. Then, we define $T_0 = \tau_v$ and for $t \geq 0$,

$$T_{2k+1} = \inf\left\{t \geq T_{2k} \,|\, v_t = w\right\} \quad \text{and}$$
$$T_{2k+2} = \inf\left\{t \geq T_{2k+1} \,|\, v_t = w\right\}.$$

The second step of the proof is somewhat classical, and consists in observing that the black-box constraints of the algorithm together with the construction of the functions f_v and f_w defined in the proof sketch of Section 7.3 imply that:

$$\begin{cases} \text{if} \quad v_t = v \quad \text{and} \quad \begin{cases} \mathcal{M}_t \supset \mathrm{Span}(\mathbf{e}_i, i \leq 2k-1) \quad \text{then} \quad \mathcal{M}_{t+1} \supset \mathrm{Span}(\mathbf{e}_i, i \leq 2k)\,, \\ \mathcal{M}_t \subset \mathrm{Span}(\mathbf{e}_i, i \leq 2k) \quad \text{then} \quad \mathcal{M}_{t+1} \subset \mathrm{Span}(\mathbf{e}_i, i \leq 2k)\,, \end{cases} \\ \text{if} \quad v_t = w \quad \text{and} \quad \begin{cases} \mathcal{M}_t \supset \mathrm{Span}(\mathbf{e}_i, i \leq 2k) \quad \text{then} \quad \mathcal{M}_{t+1} \supset \mathrm{Span}(\mathbf{e}_i, i \leq 2k+1)\,, \\ \mathcal{M}_t \subset \mathrm{Span}(\mathbf{e}_i, i \leq 2k+1) \quad \text{then} \quad \mathcal{M}_{t+1} \subset \mathrm{Span}(\mathbf{e}_i, i \leq 2k+1)\,, \end{cases} \\ \text{if} \quad v_t \notin \{v, w\}, \quad \text{then} \quad \mathcal{M}_t = \mathcal{M}_{t+1}\,. \end{cases}$$

In other words, even dimensions are discovered by node v, while odd ones are discovered by node w. The dimension $\mathbb{R}\mathbf{e}_0$ is discovered by node v thanks to the term $-b\mathbf{e}_0$. Using Theorem 1 of [CDHS21], for a right choice of parameters $\alpha, b > 0$, f is L-smooth and satisfies $f(\mathbf{x}_0) - \inf_{\mathbf{x}} f(\mathbf{x}) \leq \Delta$, together with, any k and any $\mathbf{x} \in \mathcal{M}_t \subset \mathrm{Span}(\mathbf{e}_i, i \leq 2k)$,

$$\|\nabla f(\mathbf{x})\|^2 = \frac{L\Delta}{16k^2}\,.$$

This lower bound proof technique is explained in a detailed and enlightening fashion in Chapter 3.5 of [Bub15].

Then, the final and more technical step of the proof consists in upper bounding $\mathbb{E}k(t)$. If $(T_{k+1} - T_k)_{k\geq 0}$ were independent from $k(t)$, using $\mathbb{E}\left[T_{k+1} - T_k\right] = \tau_{\text{hit}}$ for k even, we would directly obtain $t \geq \mathbb{E}\left[T_{k(t)}\right] \geq \mathbb{E}\left[k(t)\right] - 1)\tau_{\text{hit}}/2$. However, these random variables are not independent: since tail effects can happen, we need a finite second moment for hitting times,

and the proof is a bit trickier. First, note that:

$$\mathbb{E}\left[k(t)\right] = \sum_{0\leq k\leq t} \mathbb{P}\left(k(t)\geq k\right) = \sum_{0\leq k\leq t} \mathbb{P}\left(T_k \leq t\right)\,.$$

Let $(X_\ell)_{\ell\geq 0}$ be *i.i.d.* random variables of same law as τ_w conditioned on $v_0 = v$. We have $\mathbb{E}\left[X_\ell\right] = \mathbb{E}\left[\tau_w | v_0 = v\right] = \tau_{\text{hit}}$, and $\text{var}\left(X_\ell\right) < \infty$ (by assumption). Let $S_k = \sum_{\ell=0}^{k-1} X_\ell$ (S_k has the same law as $\sum_{\ell=0}^{k-1} T_{2k+1} - T_{2k}$), so that, using the Markov property of (v_t), T_k stochastically dominates $S_{\lfloor k/2\rfloor}$. Hence, $\mathbb{P}\left(T_k \leq t\right) \leq \mathbb{P}\left(S_{\lfloor k/2\rfloor} \leq t\right)$. Then, using Chebychev inequality, for any $\ell \geq 0$ and for t such that $\ell\tau_{\text{hit}} \geq t$, we have:

$$\begin{aligned}
\mathbb{P}\left(S_\ell \leq t\right) &= \mathbb{P}\left(S_\ell - \ell\tau_{\text{hit}} \leq t - \ell\tau_{\text{hit}}\right) \\
&= \mathbb{P}\left((S_\ell - \ell\tau_{\text{hit}})^2 \leq (t - \ell\tau_{\text{hit}})^2\right) \\
&\leq \frac{\ell \text{var}\left(X_0\right)}{(t-\ell\tau_{\text{hit}})^2}\,.
\end{aligned}$$

We then have:

$$\begin{aligned}
\mathbb{E}\left[k(t)\right] &\leq 2\sum_{0\leq\ell\leq t/2} \mathbb{P}\left(S_\ell \leq t\right) \\
&= 2\sum_{0\leq\ell\leq 2t/\tau_{\text{hit}}} \mathbb{P}\left(S_\ell \leq t\right) + 2\sum_{2t/\tau_{\text{hit}}\leq\ell\leq t/2} \mathbb{P}\left(S_\ell \leq t\right) \\
&\leq \frac{4t}{\tau_{\text{hit}}} + 2\sum_{2t/\tau_{\text{hit}}\leq\ell\leq t/2} \frac{\ell \text{var}\left(X_0\right)}{(t-\ell\tau_{\text{hit}})^2}\,.
\end{aligned}$$

We finally show that the second term stays bounded:

$$\begin{aligned}
\sum_{2t/\tau_{\text{hit}}\leq\ell\leq t/2} \frac{\ell}{(t-\ell\tau_{\text{hit}})^2} &= \frac{1}{\tau_{\text{hit}}^2} \sum_{0\leq\ell\leq t/2-2t/\tau_{\text{hit}}} \frac{\ell + 2t/\tau_{\text{hit}}}{(\ell + t/\tau_{\text{hit}})^2} \\
&= \frac{1}{\tau_{\text{hit}}^2} \sum_{0\leq\ell\leq t/2-2t/\tau_{\text{hit}}} \frac{\ell}{(\ell + t/\tau_{\text{hit}})^2} + \frac{2}{\tau_{\text{hit}}^2} \sum_{0\leq\ell\leq t/2-2t/\tau_{\text{hit}}} \frac{t/\tau_{\text{hit}}}{(\ell + t/\tau_{\text{hit}})^2}\,.
\end{aligned}$$

First, using a comparison with a continuous sum, we have:

$$\sum_{0\leq\ell\leq t} \frac{\ell}{(\ell + t/\tau_{\text{hit}})^2} \leq \sum_{0\leq\ell\leq t} \frac{1}{(\ell + t/\tau_{\text{hit}})} \leq \ln(\frac{t}{t/\tau_{\text{hit}}}) = \ln(\tau_{\text{hit}})\,,$$

since for $a, x > 0$, $\int_0^{ax} \frac{y\text{d}y}{(y+a)^2} \leq \int_0^{ax} \frac{\text{d}y}{(y+a)} = \ln(x)$. Finally, using $\sum_{\ell\geq 1} \frac{1}{(a+\ell)^2} \leq \int_0^\infty \frac{\text{d}y}{(y+a)^2} = \frac{1}{a}$, we bound the second sum as:

$$\sum_{0\leq\ell\leq t/2-2t/\tau_{\text{hit}}} \frac{t/\tau_{\text{hit}}}{(\ell + t/\tau_{\text{hit}})^2} \leq \frac{\tau_{\text{hit}}}{t} + 1\,.$$

Wrapping our arguments together, we end up with:

$$\mathbb{E}\left[k(t)\right] \leq \frac{4t}{\tau_{\text{hit}}} + \frac{2\text{var}\left(\tau_v\right)}{\tau_{\text{hit}}^2}(\ln(\tau_{\text{hit}}) + 1 + \frac{\tau_{\text{hit}}}{t})\,.$$

For t big enough, we end up with $\mathbb{E}\left[k(t)\right] \leq 5t/\tau_{\text{hit}}$, so that since $\mathbb{E}\|\nabla f(\mathbf{x}_t)\|^2 \geq L\Delta/(16\mathbb{E}\left[k(t)\right]^2)$

as explained in the main text, we have:

$$\|\nabla f(\mathbf{x}_t)\|^2 = \Omega\left(\frac{L\Delta\tau_{\text{hit}}^2}{t^2}\right).$$

□

7.B. Markov chain stochastic gradient descent: proof of Theorem 7.3

The following proofs in this Appendix section are valid for finite as well as infinite state spaces $\mathcal{V}$.

We start by proving the following bound on $\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}_t)\|^2\right]$. Note that this bound can be used for any $t \geq \tau_{\text{mix}}$.

Lemma 7.B.1. *For $t \geq 0$ and if $v_t \sim \pi_t$ for $\mathrm{d}_{\text{TV}}(\pi_t, \pi) \leq \pi_{\min}/2$, we have:*

$$\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}_t)\|^2\right] \leq 3\bar{\sigma}^2 + 2\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right].$$

Proof of the Lemma. We have for any $v \in \mathcal{V}$ that $\mathbb{P}(v_t = v) \leq \pi_v + \pi_v/2 = 3\pi_v/2$, so that

$$\begin{aligned}\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}_t)\|^2\right] &\leq 2\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}_t) - \nabla f(\mathbf{x}_t)\|^2\right] + 2\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] \\ &\leq 2\sum_{v\in\mathcal{V}} \mathbb{P}(v_t = v)\,\sigma_v^2 + 2\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] \\ &= 3\bar{\sigma}^2 + 2\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right].\end{aligned}$$

□

The proof borrows ideas from both the analyses of delayed SGD [MPP+17] and SGD with biased gradients [EMS22b], thus refining MC-SGD initial analysis [JRJ10]. While a biased gradient analysis would not yield convergence to an ε-stationary point for arbitrary ε (at every iterations, biases are non-negligible and can be arbitrary high), by enforcing a delay τ (of order τ_{mix}) in the analysis, we manage to take advantage of the ergodicity of the biases.

7.B.1. Smooth non-convex case of Theorem 7.3

Proof of Theorem 7.3.1. Denoting $F_t = \mathbb{E}f(\mathbf{x}_t) - f(\mathbf{x}^\star)$, we have using smoothness:

$$F_{t+1} - F_t \leq -\gamma\mathbb{E}\left[\langle\nabla f_{v_t}(\mathbf{x}_t), \nabla f(\mathbf{x}_t)\rangle\right] + \frac{\gamma^2 L}{2}\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}_t)\|^2\right].$$

For the first term on the righthandside of the inequality, assuming that $t \geq \tau$ for some $\tau > 0$ we explicit later in the proof:

$$\begin{aligned}\mathbb{E}\left[-\gamma\langle\nabla f_{v_t}(\mathbf{x}_t), \nabla f(\mathbf{x}_t)\rangle\right] &= \mathbb{E}\left[-\gamma\langle\nabla f_{v_t}(\mathbf{x}_{t-\tau}), \nabla f(\mathbf{x}_{t-\tau})\rangle\right] + \mathbb{E}\left[-\gamma\langle\nabla f_{v_t}(\mathbf{x}_t), \nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-\tau})\rangle\right] \\ &\quad + \mathbb{E}\left[-\gamma\langle\nabla f_{v_t}(\mathbf{x}_t) - \nabla f_{v_t}(\mathbf{x}_{t-\tau}), \nabla f(\mathbf{x}_{t-\tau})\rangle\right].\end{aligned}$$

First, we condition the first term on the filtration up to time $t-\tau$:

$$\begin{aligned}\mathbb{E}\left[-\gamma\langle\nabla f_{v_t}(\mathbf{x}_{t-\tau}), \nabla f(\mathbf{x}_{t-\tau})\rangle\right] &= \mathbb{E}\left[-\gamma\langle\mathbb{E}_{t-\tau}\nabla f_{v_t}(\mathbf{x}_{t-\tau}), \nabla f(\mathbf{x}_{t-\tau})\rangle\right] \\ &\leq -\frac{\gamma}{2}\mathbb{E}\left[\left\|\mathbb{E}_{t-\tau}\nabla f_{v_{t-\tau}}(\mathbf{x}_t)\right\|^2\right] + \frac{\gamma}{2}\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau}) - \mathbb{E}_{t-\tau}\nabla f_{v_t}(\mathbf{x}_{t-\tau})\|\right] \\ &\quad - \frac{\gamma}{2}\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right].\end{aligned}$$

Then, for $\tau \geq \tau_{\text{mix}}(\pi_{\min}\varepsilon)$, using the following lemma, we have, for $\varepsilon < 1/2$:

$$\mathbb{E}\left[-\gamma\langle \mathbb{E}_{t-\tau}\nabla f_{v_t}(\mathbf{x}_{t-\tau}), \nabla f(\mathbf{x}_{t-\tau})\rangle\right] \leq -\frac{\gamma}{4}\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + \gamma\varepsilon^2\bar{\sigma}^2\,.$$

Lemma 7.B.2. *For $\tau \geq \tau_{\text{mix}}(\varepsilon\pi_{\min})$ and $t \geq \tau$,*

$$\mathbb{E}\left[\|\mathbb{E}_{t-\tau}\nabla f_{v_t}(\mathbf{x}_{t-\tau}) - \nabla f(\mathbf{x}_{t-\tau})\|^2\right] \leq 2\varepsilon^2\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + 2\varepsilon^2\bar{\sigma}^2\,.$$

Proof of the Lemma. We have:

$$\begin{aligned}\mathbb{E}\left[\|\mathbb{E}_{t-\tau}\nabla f_{v_t}(\mathbf{x}_{t-\tau}) - \nabla f(\mathbf{x}_{t-\tau})\|^2\right] &= \mathbb{E}\left[\left\|\sum_{v\in\mathcal{V}}(\mathbb{P}\left(v_t = v|\mathbf{x}_{t-\tau}\right) - \pi_v)\nabla f_v(\mathbf{x}_{t-\tau})\right\|^2\right] \\ &\leq \varepsilon^2\sum_{v\in\mathcal{V}}\pi_v\mathbb{E}\left[\|\nabla f_v(\mathbf{x}_{t-\tau})\|^2\right],\end{aligned}$$

where we used $|\mathbb{P}\left(v_t = v|\mathbf{x}_{t-\tau}\right) - \pi_v| \leq \varepsilon\pi_v$ and convexity of the squared Euclidean norm. For that last term,

$$\begin{aligned}\sum_{v\in\mathcal{V}}\pi_v\mathbb{E}\left[\|\nabla f_v(\mathbf{x}_{t-\tau})\|^2\right] &\leq \sum_{v\in\mathcal{V}}2\pi_v\left(\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + \sigma_v^2\right) \\ &= 2\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + 2\bar{\sigma}^2\,,\end{aligned}$$

concluding the proof of the Lemma. □

Using gradient Lipschitzness and writing $\mathbf{x}_t - \mathbf{x}_{t-\tau} = -\gamma\sum_{s=\max(t-\tau,0)}^{t-1}\nabla f_{v_s}(\mathbf{x}_s)$, we have:

$$\begin{aligned}\mathbb{E}\left[-\gamma\langle\nabla f_{v_t}(\mathbf{x}_t), \nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-\tau})\rangle\right] &\leq \gamma^2 L\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}_t)\|\left\|\sum_{s=\max(t-\tau,0)}^{t-1}\nabla f_{v_s}(\mathbf{x}_s)\right\|\right] \\ &\leq \frac{\gamma^2 L}{2}(\tau\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}_t)\|^2\right] + \sum_{s=\max(t-\tau,0)}^{t-1}\mathbb{E}\left[\|\nabla f_{v_s}(\mathbf{x}_s)\|^2\right]).\end{aligned}$$

Similarly,

$$\mathbb{E}\left[-\gamma\langle\nabla f_{v_t}(\mathbf{x}_t) - \nabla f_{v_t}(\mathbf{x}_{t-\tau}), \nabla f(\mathbf{x}_{t-\tau})\rangle\right] \leq \frac{\gamma^2 L}{2}(\tau\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + \sum_{s=\max(t-\tau,0)}^{t-1}\mathbb{E}\left[\|\nabla f_{v_s}(\mathbf{x}_s)\|^2\right]).$$

Wrapping things up, we obtain, for $t \geq \tau$ and $\tau \geq \tau_{\text{mix}}$:

$$\begin{aligned}F_{t+1} - F_t &\leq -\frac{\gamma}{4}\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + \gamma\varepsilon^2\bar{\sigma}^2 \\ &\quad + \frac{\gamma^2 L}{2}\big((\tau+1)\mathbb{E}\left[\|\nabla f_{v_t}(\mathbf{x}_t)\|^2\right] + \tau\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + 2\sum_{s=\max(t-\tau,0)}^{t-1}\mathbb{E}\left[\|\nabla f_{v_s}(\mathbf{x}_s)\|^2\right]\big) \\ &\leq -\frac{\gamma}{4}\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + \gamma\varepsilon^2\bar{\sigma}^2 + (3\tau+1)\frac{\gamma^2 L}{2} \\ &\quad + \frac{\gamma^2 L}{2}\big((\tau+1)\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] + \tau\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + 2\sum_{s=\max(t-\tau,0)}^{t-1}\mathbb{E}\left[\|\nabla f(\mathbf{x}_s)\|^2\right]\big)\,.\end{aligned}$$

Summing for $\tau \leq t < T$:

$$\frac{1}{T}\sum_{\tau\leq t<T}\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] \leq \frac{4F_\tau}{\gamma T}+\frac{1}{T}\sum_{\tau\leq t<T}6\gamma L\tau\big(\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right]+\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right]\big)+2\big(2\varepsilon^2+\gamma(3\tau+1)\big)\bar{\sigma}^2\,,$$

leading to, for $\gamma \leq \frac{1}{12L\tau}$:

$$\frac{1}{T}\sum_{t<T-\tau}\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] \leq \frac{4F_\tau}{\gamma T}+\frac{6\gamma L\tau}{T}\sum_{T-\tau\leq t<T}\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right]+2\big(2\varepsilon^2+\gamma(3\tau+1)\big)\bar{\sigma}^2\,. \tag{7.8}$$

We now prove that for any $t \geq 0$, we have $\sup_{t\leq s\leq t+\tau}\mathbb{E}\left[\|\nabla f(\mathbf{x}_s)\|^2\right] \leq 4\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] + 8\gamma^2L^2\tau^2\sigma^2$. Let $t \leq s < t+\tau$.

$$\begin{aligned}
\mathbb{E}\left[\|\nabla f(\mathbf{x}_s)\|^2\right] &\leq 2\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right]+2\mathbb{E}\left[\|\nabla f(\mathbf{x}_s)-\nabla f(\mathbf{x}_t)\|^2\right]\\
&\leq 2\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right]+2L^2\gamma^2\mathbb{E}\left[\sum_{r=t}^{s-1}\|\nabla f_{v_r}(\mathbf{x}_r)\|^2\right]\\
&\leq 2\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right]+4L^2\gamma^2\tau\sum_{r=t}^{s-1}\mathbb{E}\left[\|\nabla f(\mathbf{x}_r)\|^2\right]+\bar{\sigma}^2\\
&\leq 2\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right]+4L^2\gamma^2\tau^2\big(\sup_{t\leq s\leq t+\tau}\mathbb{E}\left[\|\nabla f(\mathbf{x}_s)\|^2\right]+\bar{\sigma}^2\big)\,,
\end{aligned}$$

leading to the desired result for $\gamma \leq 1/(8L\tau)$. Plugging this in (7.8):

$$\begin{aligned}
\frac{1}{T}\sum_{t<T-\tau}\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] \leq \frac{4F_\tau}{\gamma T}+\frac{24\gamma L\tau}{T}\sum_{T-\tau\leq t<T}\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right]\\
+\frac{\tau}{T}4L^2\gamma^2\tau^2\bar{\sigma}^2+2\big(2\varepsilon^2+\gamma(3\tau+1)\big)L\bar{\sigma}^2\,.
\end{aligned}$$

Now, for $\gamma = \min(1/(48L\tau), \sqrt{F_0/(TL\tau\bar{\sigma}^2)})$, $\varepsilon = 1/\sqrt{T}$, and so $\tau = \tau_{\text{mix}}\ln(T)$, we have:

$$\frac{1}{T}\sum_{t<T-\tau}\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] \leq \frac{196\tau LF_\tau}{T}+7\sqrt{\frac{LF_0\bar{\sigma}^2}{T}}\,. \tag{7.9}$$

We now upper bound F_τ. For any t and $\gamma < 1/(2L)$,

$$F_{t+1}-F_t \leq \frac{\gamma}{2}\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)-\nabla f_{v_t}(\mathbf{x}_t)\|^2\right] \leq \gamma\sigma_{\max}^2/2\,,$$

where the first inequality is a simplified version of the descent lemma with biased gradient at the beggining of this proof, and the second inequality uses the initialization properties of v_0. Thus, we obtain $F_\tau \leq F_0+\gamma\tau\sigma_{\max}^2/2 \leq F_0+\sigma_{\max}^2/L$ for our choice of γ. We thus conclude by plugging this in (7.9) applied for $T+\tau$ instead of T, yielding the desired result.

The condition for the upper-bound we proved above to be true, namely $T \geq \tau = \tau_{\text{mix}}\ln(T)$, is always satisfied for $T \geq 2\tau_{\text{mix}}\ln(\tau_{\text{mix}})$. Indeed, if $T \leq \tau_{\text{mix}}^2$, then $\tau_{\text{mix}}\ln(T) \leq 2\tau_{\text{mix}}\ln(\tau_{\text{mix}}) \leq T$, and otherwise we have $\tau_{\text{mix}}\ln(T) \leq \sqrt{T}\ln(T) \leq T$. This concludes the proof, and $\tilde{\mathbf{x}}_0$ in the Theorem corresponds to $\mathbf{x}_\tau$. □

7.B.2. Under a μ-PL inequality

Proof of Theorem 2.2. We start from:

$$F_{t+1} - F_t \leq -\frac{\gamma}{4}\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + \gamma\varepsilon^2\bar{\sigma}^2 + (3\tau+1)\frac{\gamma^2 L}{2} + \frac{\gamma^2 L}{2}\Big((\tau+1)\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] + \tau\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + 2\sum_{s=\max(t-\tau,0)}^{t-1}\mathbb{E}\left[\|\nabla f(\mathbf{x}_s)\|^2\right]\Big).$$

If f satisfies a μ-PL inequality, then $-\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] \leq -2\mu F_{t-\tau}$, so that, for some $\alpha \in (0,1)$:

$$F_{t+1} - F_t \leq -\frac{\alpha\gamma\mu}{4}F_{t-\tau} - \frac{(1-\alpha)\gamma}{8}\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + \gamma\varepsilon^2\bar{\sigma}^2 + (3\tau+1)\frac{\gamma^2 L}{2}\bar{\sigma}^2 + \frac{\gamma^2 L}{2}\Big((\tau+1)\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right] + \tau\mathbb{E}\left[\|\nabla f(\mathbf{x}_{t-\tau})\|^2\right] + 2\sum_{s=\max(t-\tau,0)}^{t-1}\mathbb{E}\left[\|\nabla f(\mathbf{x}_s)\|^2\right]\Big).$$

For $t \geq 0$, let $P_t = (1-\alpha\gamma\mu/4)^{-t}$. We multiply the above expression by P_{t+1} and sum for $t < T$, hoping for cancellations. For $T \geq \tau$:

$$\begin{aligned}
\sum_{\tau\leq t<T} P_{t+1}\big(F_t - F_{t+1} - \frac{\alpha\gamma\mu}{4}F_{t-\tau}\big) &= \sum_{\tau\leq t<T} P_{t+1}\big((1-\frac{\alpha\gamma\mu}{4})F_t - F_{t+1} + \frac{\alpha\gamma\mu}{4}(F_t - F_{t-\tau})\big) \\
&= \sum_{\tau\leq t<T} P_t F_t - \sum_{\tau+1\leq t\leq T} P_t F_t \\
&\quad + \frac{\alpha\gamma\mu}{4}\sum_{\tau\leq t<T} P_{t+1}F_t - \frac{\alpha\gamma\mu}{4}\sum_{\tau\leq t<T} P_{t+1}F_{t-\tau} \\
&\leq P_\tau F_\tau - P_T F_T + \frac{\alpha\gamma\mu}{4}\sum_{\tau\leq t<T} P_{t+1}F_t - \frac{P_\tau\alpha\gamma\mu}{4}\sum_{0\leq t<T-\tau} P_{t+1}F_t \\
&\leq P_\tau F_\tau - P_T F_T + \frac{\alpha\gamma\mu}{4}\sum_{T-\tau\leq t<T} P_{t+1}F_t \\
&\leq P_\tau F_\tau - P_T F_T + \frac{\alpha\gamma}{8}\sum_{T-\tau\leq t<T} P_{t+1}\mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right],
\end{aligned}$$

using the μ-PL inequality. For $t \geq 0$, we denote $R_t = \mathbb{E}\left[\|\nabla f(\mathbf{x}_t)\|^2\right]$. We now handle the "$R_t$" terms.

$$\begin{aligned}
-\sum_{\tau\leq t<T}\frac{(1-\alpha)\gamma}{8}P_{t+1}R_{t-\tau} + \sum_{\tau\leq t<T}\frac{\gamma^2 L}{2}\big((\tau+1)P_{t+1}R_t + \tau P_{t+1}R_{t-\tau} + 2\sum_{s=t-\tau}^{t-1}P_{t+1}R_s\big) \\
\leq -\sum_{0\leq t<T-\tau}\frac{(1-\alpha)\gamma}{8}P_{t+\tau+1}R_t \\
+ \frac{\gamma^2 L}{2}\left((\tau+1)\sum_{\tau\leq t<T}P_{t+1}R_t + \tau\sum_{0\leq t<T-\tau}P_{t+1}R_t + 2\tau\sum_{t<T}R_t P_{t+\tau}\right) \\
= -\sum_{0\leq t<T-\tau}P_{t+1}R_t\gamma\Big(\frac{(1-\alpha)}{8}P_\tau - \frac{\gamma L}{2}(2\tau+1+2\tau P_{\tau-1})\Big)
\end{aligned}$$

$$
\begin{aligned}
&+ \frac{\gamma^2 L}{2} \sum_{T-\tau \le t < T} \big((\tau + 1 + 2\tau P_{\tau-1}) \big) P_{t+1} R_t \\
&\le - \sum_{0 \le t < T-\tau} \frac{(1-\alpha)\gamma}{16} P_{t+\tau+1} R_t \\
&+ \quad \frac{(1-\alpha)\gamma}{16\beta} \sum_{T-\tau \le t < T} P_{t+1} R_t \,,
\end{aligned}
$$

if γ satisfies $\gamma \le \frac{1-\alpha}{8\beta L(5\tau+1)}$ and $P_\tau \le 2$, for some $\beta \ge 1$. Since for $\gamma\mu \le 1$, $P_\tau \le e^{\tau\mu\gamma}$, $P_\tau \le 2$ can be ensured with $\gamma \le \frac{1}{2\tau L}$. All in one, we have:

$$
\begin{aligned}
0 \le P_\tau F_\tau - P_T F_T &+ \gamma \Big(\frac{\alpha}{8} + \frac{1-\alpha}{16\beta} \Big) \sum_{T-\tau \le t < T} P_{t+1} R_t \\
&- \sum_{0 \le t < T-\tau} \frac{(1-\alpha)\gamma}{16} P_{t+\tau+1} R_t \\
&+ \Big(\gamma \varepsilon^2 \bar{\sigma}^2 + (3\tau+1) \frac{\gamma^2 L}{2} \bar{\sigma}^2 \Big) \sum_{\tau \le t < T} P_{t+1} \,.
\end{aligned}
$$

Using what we proved in the previous proof, we have $R_t \le 4R_{t-\tau} + 8\gamma^2 L^2 \tau^2 \sigma^2$ for $T - \tau \le t < T$, so that:

$$
\begin{aligned}
\gamma \Big(\frac{\alpha}{8} + \frac{1-\alpha}{16\beta} \Big) \sum_{T-\tau \le t < T} P_{t+1} R_t \le 4\gamma \Big(\frac{\alpha}{8} + \frac{1-\alpha}{16\beta} \Big) &\sum_{T-2\tau \le t < T-\tau} P_{t+\tau+1} R_t \\
&+ 8\gamma^2 L^2 \tau^2 \sigma^2 \gamma \Big(\frac{\alpha}{8} + \frac{1-\alpha}{16\beta} \Big) \sum_{T-2\tau \le t < T-\tau} P_{t+\tau+1} \,.
\end{aligned}
$$

Consequently, for $4\gamma \Big(\frac{\alpha}{8} + \frac{1-\alpha}{16\beta} \Big) \le \frac{1-\alpha}{16}\gamma$, which can be ensured with $\alpha = 1/16$ and $\beta = 8$, we have:

$$
\begin{aligned}
0 \le P_\tau F_\tau - P_T F_T &+ \frac{1}{8} \gamma^3 L^2 \tau^2 \sigma^2 \sum_{T-2\tau \le t < T-\tau} P_{t+\tau+1} \\
&+ \Big(\gamma \varepsilon^2 + (3\tau+1) \frac{\gamma^2 L}{2} \Big) \bar{\sigma}^2 \sum_{\tau \le t < T} P_{t+1} \,,
\end{aligned}
$$

so that:

$$
\begin{aligned}
F_T &\le F_\tau / P_{T-\tau} + \gamma^2 \bar{\sigma}^2 L \Big(\frac{\varepsilon^2}{L\gamma} + \frac{3\tau+1}{2} + \frac{\gamma L \tau^2}{8} \Big) \frac{\sum_{t \le T} P_t}{P_T} \\
&\le 2F_\tau / P_T + \frac{2\gamma\bar{\sigma}^2}{\mu} L \Big(\frac{\varepsilon^2}{L\gamma} + \frac{3\tau+1}{2} + \frac{\gamma L \tau^2}{8} \Big) \\
&\le 2F_\tau / P_T + \frac{2\gamma\bar{\sigma}^2}{\mu} L \big(\frac{\varepsilon^2}{L\gamma} + \frac{1}{2} + 2\tau \big) \,.
\end{aligned}
$$

Finally, using $F_\tau \le F_0 + \sigma_{\max}^2 / L$, if $\gamma \le \frac{1}{64(5\tau+1)}$ where $\tau = \tau_{\mathrm{mix}}(\varepsilon)$:

$$
F_T \le 2(F_0 + \sigma_{\max}^2 / L)(1 - \frac{\gamma\mu}{8})^T + \frac{2\gamma\bar{\sigma}^2}{\mu} \big(\frac{\varepsilon^2}{L\gamma} + \frac{1}{2} + 2\tau \big)
$$

We thus choose $\varepsilon = \sqrt{1/T}$ so that $\tau \leq \tau_{\text{mix}} \ln(T)$, and stepsize $\gamma = \min(\frac{8\ln(T(F_0+\bar{\sigma}^2)/\bar{\sigma}^2)}{\mu T}, \frac{1}{64(5\tau+1)})$, leading to the desired result for $c = 64 \times 6 = 384$. The same discussion than in the smooth non-convex proof regarding the condition $T \geq \tau$ applies here. □

7.B.3. Proof of Proposition 7.4.1

Proof. Consider the graph G on the set of nodes $\mathcal{V} = \{0, 1\}$, with probability transitions $p_{01} = p_{10} = p$ and $p_{00} = p_{11} = 1-p$, for some small $p \in (0, 1)$. The relaxation time $\tau_{\text{mix}}(1/4)$ of this graph scales as $1/p$.

Consider now $f_0(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-1)^2$ and $f_1(\mathbf{x}) = \frac{1}{2}(\mathbf{x}+1)^2$ for $\mathbf{x} \in \mathbb{R}$, so that $\mathbf{x}^\star = 0$. For (v_t) a Markov chain with the given transition probabilities, started at v_0 following the uniform (stationary) distribution on $\mathcal{V}$, let $\mathbf{x}_t$ be generated with MC-SGD: $\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma \nabla f_{v_t}(\mathbf{x}_t)$ and $\mathbf{x}_0 = 1$, *i.e.*,

$$\mathbf{x}_T = (1-\gamma)^T - \gamma \sum_{t<T} (1-\gamma)^{T-t-1} \zeta_t \,,$$

where $\zeta_t \in \{-1, 1\}$ takes value 1 if $v_t = 0$ and value -1 if $v_t = 0$. We have:

$$\mathbb{E}\left[(\mathbf{x}_T - \mathbf{x}^\star)^2\right] = (1-\gamma)^{2T} + \gamma^2 \sum_{s<t<T} (1-\gamma)^{2T-t-s-2} \mathbb{E}\left[\zeta_s \zeta_t\right].$$

We compute this second term, and show that it is non-negative for $p \leq 1/2$ and of order $\frac{\gamma}{4p}$, so that to reach a given precision $\varepsilon > 0$, is required $\gamma \leq 4p\varepsilon$ and thus to make the first term small, T must verify $T = \Omega(1/(2p\varepsilon))$, concluding our reasonning.

For $s < t$, we have $\mathbb{E}\left[\zeta_s \zeta_t\right] = 2\mathbb{P}\left(v_{t-s} = v_0 | v_0 \sim \pi^\star\right) - 1 /$ Denoting $z_k = \mathbb{P}\left(v_k = v_0 | v_0 \sim \pi^\star\right)$, we have $z_{k+1} = p z_k + (1-p)(1-z_k)$ and $z_0 = 1$, so that $z_k = \frac{1}{2}(1 + (1-2p)^k)$ for $k \geq 0$. This leads to:

$$\gamma^2 \sum_{s<t<T} (1-\gamma)^{2T-t-s-2} \mathbb{E}\left[\zeta_s \zeta_t\right] = \gamma^2 \frac{(1-\gamma)(1-2p)}{1-(1-\gamma)(1-2p)} \times \left(\frac{1-(1-\gamma)^{2T}}{1-(1-\gamma)^2} - (1-2p) \frac{(1-\gamma)^T - (1-2p)^T}{2p-\gamma} (1-\gamma)^T \right)$$

For $\varepsilon \to 0$, in order to have $\mathbb{E}\left[(\mathbf{x}_T - \mathbf{x}^\star)^2\right] \leq \varepsilon$, is required $(1-\gamma)^T \leq \varepsilon$ so that $\gamma T \to \infty$. Under $\gamma T \to \infty$, we have

$$\gamma^2 \sum_{s<t<T} (1-\gamma)^{2T-t-s-2} \mathbb{E}\left[\zeta_s \zeta_t\right] \sim \frac{\gamma}{4p} \,.$$

Finally, to reach precision ε, this quantity needs to be upper-bounded by $\varepsilon(1+o(1))$, so that $\gamma \leq 4p\varepsilon^{-1}(1+o(1))$ is necessary. Plugging this in $(1-\gamma)^{2T} \leq \varepsilon$ yields $T = \tilde{\Omega}(p\varepsilon^{-1})$, the desired result. □

7.C. With local noise: proof of Theorem 7.5

The proof follows the exact same steps as the proof of Theorem 7.4.

Proof. First, note that, using Lemma 15 in [SK20], we have:

$$\mathbb{E}\left[\left\|\sum_{t<T} \nabla_{\mathbf{x}} F_{v_t}(\mathbf{x}^\star, \xi_t)\right\|^2\right] \leq 2\mathbb{E}\left[\left\|\sum_{t<T} \nabla f_{v_t}(\mathbf{x}^\star)\right\|^2\right] + 2T\sigma_{\text{local}}^2 \,.$$

We then have the following lemma, proved exactly as in the previous section.

Lemma 7.C.1. *For any $\mathbf{y}_t \in \mathbb{R}^d$ and $t \geq 0$, denoting $\mathbf{y}_{t+1} = \mathbf{y}_t - \gamma \nabla_{\mathbf{x}} F_{v_t}(\mathbf{x}^\star, \xi_t)$, we have:*

$$\|\mathbf{x}_{t+1} - \mathbf{y}_{t+1}\|^2 \leq (1 - \gamma\mu)\|\mathbf{x}_t - \mathbf{y}_t\|^2 + \gamma L \|\mathbf{y}_t - \mathbf{x}^\star\|^2 .$$

This leads to:

$$\|\mathbf{x}_T - \mathbf{y}_T\|^2 \leq (1 - \gamma\mu)^T \|\mathbf{x}_0 - \mathbf{y}_0\|^2 + \gamma L \sum_{t<T} (1 - \gamma\mu)^{T-t} \|\mathbf{y}_t - \mathbf{x}^\star\|^2 ,$$

for

$$\mathbf{y}_0 = \mathbf{x}^\star + \gamma \sum_{t<T} \nabla_{\mathbf{x}} F_{v_t}(\mathbf{x}^\star, \xi_t) , \quad \mathbf{y}_s = \mathbf{x}^\star + \gamma \sum_{s \leq t < T} \nabla_{\mathbf{x}} F_{v_t}(\mathbf{x}^\star, \xi_t) , \quad s < T .$$

We thus have:

$$\mathbb{E}\|\mathbf{x}_T - \mathbf{x}^\star\|^2 \leq 2(1 - \gamma\mu)^T \left(\mathbb{E}\|\mathbf{x}_0 - \mathbf{x}^\star\|^2 + \gamma^2 \mathbb{E}\left[\left\| \sum_{s<T} \nabla_{\mathbf{x}} F_{v_s}(\mathbf{x}^\star, \xi_s) \right\|^2 \right] \right)$$
$$+ \gamma^3 L \sum_{t<T} (1 - \gamma\mu)^{T-t} \mathbb{E}\left[\left\| \sum_{t \leq s < T} \nabla_{\mathbf{x}} F_{v_s}(\mathbf{x}^\star, \xi_s) \right\|^2 \right] .$$

To conclude, we use the first inequality of this proof, and Lemma 7.4.2, and proceed as in the proof of Theorem 7.4 and Corollary 7.4.1. □

Chapter 8

Sample optimality in personalized collaborative learning

In personalized federated learning, each member of a potentially large set of agents aims at training a model minimizing its loss function averaged over its local data distribution. In this chapter, we study this problem under the lens of stochastic optimization, focusing on a scenario with a large number of agents, that can each only access very few data samples from their local data distribution. Specifically, we prove novel matching lower and upper bounds on the number of samples required from all agents to approximately minimize the generalization error of a fixed agent. We provide strategies matching these lower bounds, based on a *gradient filtering* approach: given prior knowledge on some notion of distance between local data distributions, agents filter and aggregate stochastic gradients received from other agents, in order to achieve an optimal bias-variance trade-off. Finally, we quantify the impact of using rough estimations of the distances between local distributions of agents, based on a very small number of local samples.

8.1. Introduction

An ideal approach for personalization would take the best of both worlds: increased statistical efficiency by using more samples, while keeping local generalization errors low. This raises the fundamental question: what is the optimal bias/variance tradeoff between personalization and coordination, and how can it be achieved?

We formulate the personalized federated learning problem, studying it under the lens of stochastic optimization [BCN18]. Consider $n \in \mathbb{N}^*$ agents denoted by integers $1 \leq i \leq n$, each desiring to minimize its own local function $f_i : \mathbb{R}^d \to \mathbb{R}$, while sharing their stochastic gradients. Since only a limited number of samples are locally available, we focus on *stochastic gradient descent*-like algorithms, where agents each sequentially compute stochastic gradients g_i^k such that $\mathbb{E}\left[g_i^k\right] = \nabla f_i$. In order to reduce the sample complexity, *i.e.* the number of samples or stochastic gradients required to reach small generalization error, agents thus need to use stochastic gradients from other agents, that are *biased* since in general $\mathbb{E}\left[g_i^k\right] \neq \nabla f_j$.

The algorithms we propose in this chapter are based on a *gradient filtering* approach: upon reception of stochastic gradients $(g_j^k)_j$, agent i *filters* these gradients and aggregates them using some weights λ_j into $\sum_j \lambda_j g_j^k$, in order to achieve some bias/variance trade-off.

8.1.1. Outline of the chapter

In this chapter, we consider an oracle model where at each step $k = 1, 2, \ldots$, all agents may draw a sample according to their local distribution. We aim at computing the number of stochastic gradients sampled from all agents, required to reach a small generalization error, in terms of biases (distances between functions or distributions), regularity, and noise assumptions. The oracle model, main assumptions and problem formulations are given in Section 8.2. Our main contributions are then as follows.

(i) In Section 8.3 we prove *information theoretic* lower bounds: to reach a target generalization error $\varepsilon > 0$ for a fixed agent i, no algorithm can achieve a reduction in the number of oracle calls by a factor larger than the total number of agents ε-close –in a suitable sense– to agent i.

(ii) We next study a naive strategy based on weighted gradient averaging algorithms, coined ALL-FOR-ONE, that matches this lower bound, at the cost of high communication and storage requirements.

(iii) We then propose in Section 8.5 a parallel extension of the simple weighted gradient averaging algorithm that yields an efficient algorithm for collaborative generalization error minimization problems. In this algorithm, agents compute stochastic gradients at *their* local estimate, and broadcast it to other agents who may use these to update their own estimates. For $x^k = (x_1^k, \dots, x_n^k)$ where x_i^k is the local estimate of agent i at iteration k, updates of the ALL-FOR-ALL algorithm write as:

$$x^{k+1} = x^k - \eta W g^k \,,$$

where $g^k = (g_1^k, ..., g_n^k)$ for an unbiased stochastic gradient g_i^k of function f_i, a step size η, and a carefully chosen symmetric matrix W. Agent i thus uses stochastic gradients that are doubly biased, as gradients of a "wrong function" f_j instead of f_i computed at a "wrong location" x_j^k instead of x_i^k. Interestingly, note that the ALL-FOR-ALL algorithm is not a gossip algorithm *per se* (see *e.g.* [SBB+19]), since the matrix W is not doubly-stochastic: gradients are not aggregated with weights that sum to 1. Moreover, W depends on the distance between local agents distributions, and thus requires either prior information on the local distributions, or estimating these distances as a pre-processing step.

(iv) We finally study in Section 8.6 the impact of estimating, based on a very limited number of samples, the matrix W to use in the ALL-FOR-ALL algorithm. Under a mixture model assumption on the agents, we obtain that for a bounded – up to logarithmic factors – number of samples per agent, any arbitrary small generalization can be reached, with an optimal collaboration speedup in terms of the number of agents in each mixture of the mixture model.

8.1.2. Related works

Federated Learning is a paradigm in machine learning where training is done collaboratively among several agents, taking into account privacy constraints [MMR+17, KMRR16, KMA+19, WMK+19]. A central task is the training of a common model for all agents, for which both *centralized* approaches orchestrated by a server and *decentralized* approaches with no central coordinator [NOR18] have been considered. The algorithms we propose in this chapter are well suited for a decentralized implementation.

As observed in [HHHR20], training a common model for all users can lead to poor generalization on certain tasks such as e.g. next-word prediction. To improve both accuracy and fairness, *personalized* models thus need to be learnt for each agent [LSBS20, MSS19, YBS21]. Approaches to personalization include fine-tuning [CCD21, KBT19], transfer learning techniques [TJJ20, WMK+19, DHK+21], using shared-representation models [CHMS21]. Personalization in FL can also be formulated as the training of local models with a regularization term that enforces collaboration between users [HHHR20] or with a meta-learning approach [FMO20, JKRK19, CLD+18]. We refer the interested reader to [KKP20] for a broader survey of Personalized Federated Learning.

While the goal of personalization is to minimize local generalization errors, the above cited works do not provide theoretical guarantees over the sample complexity to obtain small local errors, but instead control errors on a regularized problem, in terms of communication rounds or full gradients used, and not in terms of samples used. [DKM20, MMRS20] among others provide generalization errors under a statistical learning framework that depend on

VC-dimensions and on distances between each local data distribution and the mixture of all datasets. [DK21a, DK21b] study the bias-variance trade-off between collaboration and personalization for mean estimation in a game-theoretic framework. [CKS+21, GHKJ21, BGHJ21] also concurrently frame personalization as a stochastic optimization problem with biased gradients and are the works closest to ours. They consider the training of a single agent with biased gradients from another group of agents dedicated to this agent, and obtain performance guarantees in terms of distance between individual function f_i and the average $n^{-1}\sum_j f_j$. In contrast, we obtain more general performance bounds based on distance bounds between all pairs of functions f_i, f_j (or equivalently, pairs of local distributions), in the case where all agents desire to minimize their local objective; our "bias assumption" is also milder. In addition, we prove matching lower bounds, and study under a mixture model the statistical efficiency of our approach.

Concurrently, [DW22] use similar dissimilarity notions (Integral Probability Metrics, albeit with respect to different function spaces) to show upper-bounds that closely resemble ours of Theorem 8.3, when training over mixtures of distributions (without algorithmic solutions as the *all-for-all* algorithm), together with insights on who to collaborate with for agents. These results, obtained in a different framework than ours (hypothesis and VC-dimension bounds, rather than our stochastic optimization framework), fall into the frame of our lower bound (and match it up to constant factors), form an orthogonal line of work.

Finally, data-heterogeneity has long been a challenge in Federated optimization, as for instance noted in the analyses and performances of the Local SGD algorithm [KMR20, WPS20b]. Many algorithmic solutions have been proposed to counterweight this effect [KKM+20, MJPH21] (non-exhaustive list). Yet this line of work studies the effect of data-heterogeneity on the convergence guarantees of FL algorithms that train *one* global model, irrespectively of the local generalization property of this trained global model. Our work is orthogonal, and focuses on data-heterogeneity as a challenge for statistical meaning (local generalization) of the model(s) trained, as opposed to related works that study data-heterogeneity as a challenge in distributed or federated learning to design fast and scalable algorithms. Putting into perspective these two views on the challenge data-heterogeneity in FL seems however necessary, and stresses its importance.

8.2. Problem Statement and Assumptions

We now detail our objectives and the necessary technical assumptions. We consider general stochastic gradient methods and formulate our problem, assumptions and algorithms accordingly.

8.2.1. Problem setting

Let $\mathcal{D}_i$ for $1 \leq i \leq n$ be a probability distribution on a set Ξ (agent i's local distribution, *not* its empirical distribution), $F : \mathbb{R}^d \times \Xi \to \mathbb{R}$ a loss function. We assume that the function f_i that agent i aims at minimizing is the generalization error on agent i's local distribution:

$$f_i(x) = \mathbb{E}_{\xi_i \sim \mathcal{D}_i}\left[F(x, \xi_i)\right], \quad x \in \mathbb{R}^d. \tag{8.1}$$

We coin this problem as *collaborative generalization error minimization (GEM)*. At every iteration $k = 1, 2, \ldots$, agent i may access unbiased *i.i.d.* estimates $g_i^k(x)$ of $\nabla f_i(x)$:

$$g_i^k(x) = \nabla_x F(x, \xi_i^k), \quad \xi_i^k \sim \mathcal{D}_i, \quad x \in \mathbb{R}^d, 1 \leq i \leq n.$$

Counting the number of stochastic gradients used in the whole set of agents to reach a precision ε for f_i thus reduces to computing the number of samples required from all agents to obtain local generalization error ε for agent i. To specify the information shared between agents via access to stochastic gradients, we define the following oracle, that lets at every

iteration all agents sample a stochastic gradient. After K oracle queries, each agent will have sampled K stochastic gradients for a total of nK in the whole set of agents. Let $\left\{(\xi_1^k, \dots, \xi_n^k), k \geq 0\right\}$ a sequence of *i.i.d.* random variables of law $\mathcal{D}_1 \times \cdots \times \mathcal{D}_n$. Given the initial shared knowledge $\mathcal{S}_0$, at iterations $k = 1, 2, \dots$,

1. For all $1 \leq j \leq n$, agent j samples ξ_j^k chooses some $y_j^k \in \mathbb{R}^d$ as a $\mathcal{S}_{k-1}$-measurable function.

2. The shared memory is extended: $\mathcal{S}_k = \mathcal{S}_{k-1} \cup \left\{g_j^k(y_j^k), \xi_j^k \,,\, 1 \leq j \leq n\right\}$.

3. Agent j outputs x_j^k as a $\mathcal{S}_k$-measurable function.

For fixed target precision $\varepsilon > 0$, the objective is to find, using T_ε samples from all agents in total - corresponding to $K_\varepsilon = T_\varepsilon/n$ oracle calls -, models with local generalization error ε. Throughout this chapter, we assume that each function f_i is minimized over $\mathbb{R}^d$, and we denote by $x_i^\star$ such a minimizer. We further consider the following two standard assumptions.

Assumption 8.2.1 (Noise)**.** *There exists $\sigma^2 > 0$ such that for all $1 \leq i \leq n$ and $x \in \mathbb{R}^d$,*

$$\mathbb{E}_{\xi_i \sim \mathcal{D}_i} \|\nabla_x F(x, \xi_i) - \nabla f_i(x)\|^2 \leq \sigma^2 \,.$$

Assumption 8.2.2 (Regularity)**.** *Functions f_i are μ-strongly convex and L-smooth [Bub15].*

8.2.2. Distribution-based distances

We first introduce extensions of classical *Integral Probability Metrics* (IPMs, [SGF+10]) to multivariate functions, i.e. pseudo-distances on the set of probability measures parameterized by a set $\mathcal{H}$ of functions, fixed in the sequel.

Definition 8.2.1. *For $\mathcal{H}$ a set of functions from Ξ to $\mathbb{R}^d$ and $\mathcal{D}, \mathcal{D}'$ two probability distributions on Ξ, we define:*

$$d_{\mathcal{H}}(\mathcal{D}, \mathcal{D}') = \sup_{h \in \mathcal{H}} \left\| \mathbb{E}\left[h(\xi) - h(\xi')\right] \right\| ,$$

where $\xi \sim \mathcal{D}$ and $\xi' \sim \mathcal{D}'$. $d_{\mathcal{H}}$ is a pseudo-distance on the set of probability measures on Ξ.

This family of pseudo-distances contains a large number of standard distances between distributions, including total variation (with the set of 1-locally bounded functions, functions that send any ball of radius 1 in a ball of radius 1), the Wasserstein distance (with the set of 1-Lipschitz functions), maximum mean discrepancies (with the unit ball of a RKHS), or even a simple distance between means of the distributions (with the set of 1-Lipschitz affine functions), developed further in Section 8.6.

Assumption 8.2.3 (Distribution-based dissimilarities)**.** *For some non-negative weights $(b_{ij})_{1 \leq i,j \leq n}$, we have $d_{\mathcal{H}}(\mathcal{D}_i, \mathcal{D}_j) \leq b_{ij}$ for all $1 \leq i, j \leq n$. We further assume that either of the following holds.*

1. (Weak dissimilarities). *For all $1 \leq i \leq n$, $\left(\xi \in \Xi \mapsto \nabla_x F(x_i^\star, \xi)\right) \in \mathcal{H}$.*
2. (Strong dissimilarities). *For all $x \in \mathbb{R}^d$, $\left(\xi \in \Xi \mapsto \nabla_x F(x, \xi)\right) \in \mathcal{H}$.*

The "weak dissimilarities" assumption is of course easier to satisfy than the "strong" version, and our results will ultimately depend only on the weak assumption. Under Assumption 8.2.3 (weak version) and Assumption 8.2.2, we have

$$f_i(x_j^\star) - f_i(x_i^\star) \leq b_{ij}^2/(2\mu) \,,$$

which motivates our use of distribution-based dissimilarity assumptions.

8.3. Information-theoretic lower bound on the sample complexity

In this section, we prove lower bounds on the total number of stochastic gradients required from all agents, to reach ε-generalization for a given agent. Our lower bounds apply to collaborative GEM, i.e. functions $(f_i)_{1\leq i\leq n}$ of the form (8.1), for shared loss function F and user distributions $\mathcal{D}_1,\dots,\mathcal{D}_n$.

An oracle $\phi:\mathbb{R}^{n\times d}\to\mathcal{I}$ is a random function that answers some $\phi(x)\in\mathcal{I}$ where $\mathcal{I}$ is an information set, for every query $x\in\mathbb{R}^{n\times d}$. We adapt the definitions of [ABRW12] of sample complexity for *SGD* to our personalization problem. Formally, the *first-order* oracle we defined in Section 8.2 and that we write as $\phi\big((\mathcal{D}_i)_{i=1,\dots,n},F\big)$ for shared loss function F and user distributions $\mathcal{D}_1,\dots,\mathcal{D}_n$, returns for $x\in\mathbb{R}^{n\times d}$:

$$\phi\big((\mathcal{D}_i)_i,F\big)(x)=\Big(i,\,x_i,\,\xi_i,\,F(x_i,\xi_i),\,g_i^k(x_i)\Big)_{1\leq i\leq n},$$

where $\xi_i\sim\mathcal{D}_i$. Given distributions and a loss function $\big((\mathcal{D}_i)_i,F\big)$, we denote by $\mathbb{M}$ the set of all methods $\mathcal{M}=(\mathcal{M}_K)_{K\geq 0}$: for any $K\geq 0$, $\mathcal{M}_K$ makes K oracle calls from oracle $\phi\big((\mathcal{D}_i)_i,F\big)$ (while using $T=NK$ stochastic gradient samples from all agents), and returns $x_i^K\in\mathbb{R}^d$ for agent i as a measurable function of the K oracle calls. For a set $\mathbb{D}$ of couples of distributions and loss function $\big((\mathcal{D}_j)_j,F\big)$ defining functions $(f_i)_{1\leq i\leq n}$, we are interested in lower-bounding:

$$\inf_{\mathcal{M}\in\mathbb{M}}\sup_{((\mathcal{D}_j)_j,F)\in\mathbb{D}}\mathcal{K}_i^{\varepsilon}\Big(\mathcal{M},\big((\mathcal{D}_j)_j,F\big)\Big),$$

where $\mathcal{K}_i^{\varepsilon}\Big(\mathcal{M},\big((\mathcal{D}_j)_j,F\big)\Big)$ is the number of oracle calls required to reach generalization error $\varepsilon>0$ for agent i, and writes as:

$$\mathcal{K}_i^{\varepsilon}\Big(\mathcal{M},\big((\mathcal{D}_j)_j,F\big)\Big)=\inf\left\{K\in\mathbb{N}^*\text{ such that }\mathbb{E}\left[f_i(x_i^K)-\min_{x\in\mathbb{R}^d}f_i(x)\right]\leq\varepsilon\right\}.$$

We now define the set $\mathbb{D}$ we consider for our lower bounds. Let $b=(b_{ij})_{1\leq i,j\leq n}$ be non-negative weights that verify the triangle inequality – namely, $b_{ij}\leq b_{ik}+b_{kj}$ for all i,j,k –, and let $r,\mu,L,\sigma>0$. $\mathbb{D}_\mu^L(r,b,\sigma)$ is the set of all $\big((\mathcal{D}_i)_{1\leq i\leq n},F\big)$, such that the functions f_i parameterized by these tuples of distributions and shared loss function verify Assumptions 8.2.1, 8.2.2 and 8.2.3 for σ^2, $\mu,L>0$ and b, such that $\|x_i^\star\|\leq r$ for all $1\leq i\leq n$, and such that $f_i(x_j^\star)-f_i(x_i^\star)\leq b_{ij}^2/(2\mu)$. We use the notation $a(\cdot)=\Omega(b(\cdot))$ for $\exists C>0$ such that $a(\cdot)\geq Cb(\cdot)$.

Theorem 8.1 (IT lower bound)**.** *Let $\varepsilon\in(0,1/16)$, (b_{ij}) verifying the triangle inequality, $r,\sigma>0$. Assume that the function set $\mathcal{H}$ contains the all 1-Lipschitz affine functions and that $d_{\mathcal{H}}\leq d_{\mathrm{TV}}$. For any $i\in\{1,\dots,n\}$:*

$$\inf_{\mathcal{M}\in\mathbb{M}}\sup_{((\mathcal{D}_j)_j,F)\in\mathbb{D}_{\mu=1/r^2}^{L=1/r^2}(r,b,\sigma)}\mathcal{K}_i^{\varepsilon}\Big(\mathcal{M},\big((\mathcal{D}_j)_j,F\big)\Big)=\Omega\left(\frac{r^2\sigma^2}{\varepsilon}\times\frac{1}{\mathcal{N}_i^{\varepsilon}(\frac{b^2}{4\mu})}\right),$$

where $\mathcal{N}_i^{\varepsilon}(\frac{b^2}{4\mu})=\sum_j\mathbb{1}_{\{b_{ij}^2\leq 4\mu\varepsilon\}}$ is the number of agents j verifying $b_{ij}^2\leq\varepsilon$.

The proof of this lower bound (see below) builds on lower bounds based on Fano's inequality [DW13] for stochastic gradient descent [ABRW12] or for information limited statistical estimation [ZDJW13, DR19] that we summarize below, adapted to personalization. Theorem 8.1 states that, given the knowledge of (b_{ij}), σ^2, $\mu=L$, there exist difficult instances of the problem that satisfy all three Assumptions 8.2.1, 8.2.2 and 8.2.3, such that the number of oracle calls needed to obtain a generalization error of ε for an agent i is lower-bounded by the right hand side of the equation in Theorem 8.1.

The factor $C\sigma^2 r^2 \varepsilon^{-1}$ is reminiscent of stochastic gradient descent, and is present in [ABRW12]: without cooperation, this is the sample complexity of *SGD* for a fixed agent. Cooperation appears in the factor $1/\mathcal{N}_i^{\varepsilon}(b/(4\mu))$: the sample complexity is inversely proportional to the number of agents j that have distributions similar to that of i. One cannot hope for better than a linear *collaboration speedup* proportional to agents $4\mu\varepsilon$-close to i in terms of the distance $d_{\mathcal{H}}$. Theorem 8.1 is a *worst-case* lower bound, so that a collaboration speedup could be leveraged even for small ε, but this would require making stronger additional assumptions.

8.3.1. General framework to prove lower bounds [ABRW12]

The idea is that, when optimizing a function $f(x) = \mathbb{E}[F(x,\xi)]$ and finding a good approximation of a minimizer $x^\star$, we learn some information on the distribution $\mathcal{D}$ over which samples are drawn. In order to prove lower bounds, we construct a loss function F, and distributions $\mathcal{D}_1^\alpha, \dots, \mathcal{D}_n^\alpha$, where α is a random parameter. We argue that minimizing the objective function up to a certain precision gives a good estimator (quantified) of the random seeds α. Then, using Fano inequality, we bound the efficiency of such an estimator in terms of number of oracle calls, obtaining a lower bound on the sample complexity. This approach is inspired by [ABRW12], who prove IT-lower bounds for stochastic gradient descent. We adapt their proof technique to the personalized and multi-agent setting.

Constructing difficult loss functions For any two functions $f, g : \mathbb{R}^d \to \mathbb{R}$, we define the discrepancy measure $\rho(f,g)$ as:

$$\rho(f,g) = \inf_{x\in\mathbb{R}^d} \Big\{ f(x) + g(x) - \inf_{y\in\mathbb{R}^d} f(y) - \inf_{y\in\mathbb{R}^d} g(y) \Big\},$$

which is a pseudo metrics. Now, for a finite set $\mathcal{V}$ of parameters, let $\mathcal{G}(\delta) = \{g_\alpha^\delta,\ \alpha \in \mathcal{V}\}$ be a set of functions indexed by $\mathcal{V}$, that depend on δ (fixed in the set). The dependency in δ of each $g_\alpha \in \mathcal{G}(\delta)$ is left implicit in the following subsections. We define:

$$\psi(\delta) = \inf_{f,g\in\mathcal{G}(\delta), f\neq g} \rho(f,g)\,.$$

Minimizing is Bernoulli parameters identification The two following lemmas justify that optimizing a function $g_\alpha \in \mathcal{G}(\delta)$ to a precision of order $\psi(\delta)$ is more difficult than estimating the parameter α.

Lemma 8.3.1 ([ABRW12]). *For any $x \in \mathbb{R}^d$, there can be at most one function g_α in $\mathcal{G}(\delta)$ such that:*

$$g_\alpha(x) - \inf_{\mathbb{R}^d} g_\alpha < \frac{\psi(\delta)}{3}\,.$$

Lemma 8.3.2 ([ABRW12]). *Assume that for some fixed but unknown $\alpha \in \mathcal{V}$ there exists a method $\mathcal{M}_K$ based on the data $\phi = \{X_1, ..., X_K\}$ that returns x^K (function of ϕ) satisfying an error of:*

$$\mathbb{E}\left[g_\alpha(x^K) - \min_{x\in\mathbb{R}^d} g_\alpha(x) \right] < \frac{\psi(\delta)}{9}\,,$$

where the mean is taken over the randomness of both the oracle Φ, the method $\mathcal{M}_K$ and $\alpha \in \mathcal{V}$ if random. Then, there exists a hypothesis test $\hat{\alpha} : \phi \to \mathcal{V}$ such that:

$$\max_{\alpha\in\mathcal{V}} \mathbb{P}_\phi\big(\hat{\alpha} \neq \alpha\big) \leq \frac{1}{3}\,.$$

Suppose now that the parameter α in the previous Lemma is chosen uniformly at random

in $\mathcal{V}$. Let $\hat{\alpha} : \phi \to \mathcal{V}$ be a hypothesis test estimating α. By Fano inequality [CT05], we have:

$$\mathbb{P}\big(\hat{\alpha} \neq \alpha\big) \geq 1 - \frac{I\big(\phi, \alpha\big) + \ln(2)}{\ln(|\mathcal{V}|)}\,, \tag{8.2}$$

where $I\big(\phi, \alpha\big)$ is the mutual information between ϕ and α, that we need to upper-bound. Combining Fano inequality with Lemmas 8.3.1 and 8.3.2, fixing a target error $\varepsilon = \psi(\delta)$, we obtain a lower bound on the K_ε the number of oracle calls required to reach an ε generalization error:

$$\frac{1}{3} \geq \mathbb{P}_\phi\big(\hat{\alpha} \neq \alpha\big) \geq 1 - \frac{I\big(\phi_{K_\varepsilon}, \alpha\big) + \ln(2)}{\ln(|\mathcal{V}|)}\,,$$

where ϕ_{K_ε} is the information contained in K_ε oracle calls. If we have an equality of the form $I\big(\phi_{K_\varepsilon}, \alpha\big) = K_\varepsilon I\big(\phi_1, \alpha\big)$, this gives:

$$K_\varepsilon \geq \frac{\frac{2}{3}\ln(|\mathcal{V}|) - \ln(2)}{I(\phi_1, \alpha)}\,. \tag{8.3}$$

Playing with the different parameters $\delta, \alpha, \mathcal{V}$ gives lower bounds. We refer the interested reader to Chapter 2 in [CT05] for Fano inequality and mutual information.

8.3.2. Applying this to prove Theorem 8.1

Proof of Theorem 8.1. For simplicity, assume that $r^2 = d$ and $\sigma^2 = 1$. Let $\delta > 0$ a free parameter. Let $\mathcal{V} = \{\alpha^1, \dots, \alpha^L\} \subset \{-1, 1\}^d$ be a subset of the hypercube such that for all $k \neq l$,

$$\frac{1}{2}\sum_{i=1}^{d} |\alpha_i^k - \alpha_i^l| \geq \frac{d}{4}\,,$$

i.e. $\mathcal{V}$ is a $d/4$-packing of the hypercube. We know that we can set $|\mathcal{V}| \geq (2/\sqrt{e})^{d/2}$. Without loss of generality, we prove a lower bound in the case where the agent that desires to minimize its local function is indexed by 1.

Let:

$$F(x, \xi) = \frac{1}{2}\|x - \xi\|^2\,,$$

for $x, \xi \in \mathbb{R}^d$ and, for fixed $\delta > 0$ and any $\alpha \in \mathcal{V}$:

$$g_\alpha(x) = \frac{1}{2d}\sum_{k=1}^{d}\Big(x_k^2 + 1 - 2\big(\frac{1}{2} + \alpha_k\delta\big)x_k\Big) \quad , x \in \mathcal{X}\,.$$

We keep the same notations as last subsection $(\psi(\delta), \rho)$. We have:

$$\rho(g_\alpha, g_\beta) = \frac{\delta^2}{d}\sum_{k=1}^{d}|\alpha_k - \beta_k|\,,$$

leading to $\psi(\delta) \geq \delta^2/4$ since $\mathcal{V}$ is a $d/4$-packing of the hypercube.

For any $i = 1, \dots, n$, let $\mathcal{D}_i$ be the probability distribution on $\{0, 1\}^d$ of the following random variable:

$$\mathrm{Ber}\big(\frac{1}{2} + \delta_i\alpha_k\big)\epsilon_k \quad \text{where} \quad \delta_i = (\delta - b_{i1})^+\,,$$

where $s^+ = \max(0, s)$ for $s \in \mathbb{R}$, k is taken uniformly at random in $\{1, \dots, d\}$, (ϵ_k) is the canonical basis of $\mathbb{R}^d$, and $Ber(p)$ is a Bernoulli random variable, independent of k.

The mutual information is thus, in our case:

$$I(\phi_K, \alpha) \leq C_1 K \mathcal{N}_1^{\delta}(\sqrt{b})\delta^2 \,,$$

where we use the fact that $I\left(\mathrm{Ber}\left(\frac{1}{2} + \mathbb{1}_{b_{1i}\leq\delta}\alpha_k\delta_i\right), \alpha_k\right) \leq C_1\delta^2$ for some constant $C_1 > 0$, for $\delta_i \leq 1/4$. Setting the target precision as $\varepsilon = \delta^2/4$, we obtain:

$$K_\varepsilon \geq C' \frac{d}{\varepsilon \mathcal{N}_1^{\varepsilon}(4b)} \,.$$

The loss function and distributions built verify our regularity assumptions for $\mu = 1/d$, $L = 1/d$, noise $\sigma^2 \leq 1$.

We first verify that for all $1 \leq j, k \leq n$, we have $f_j(x_k^\star) - f_j(x_j^\star) \leq b_{kj}^2$. We first notice that $x_j^\star = \frac{1}{d}\left(\frac{1}{2} + \delta_j\alpha_l\right)_{1\leq l\leq d}$, so that:

$$\begin{aligned} f_j(x_k^\star) - f_j(x_j^\star) &= \frac{1}{d}\left\|x_j^\star - x_k^\star\right\|^2 \\ &= (\delta_i - \delta_k)^2 \\ &\leq (b_{1j} - b_{1k})^2 \\ &\leq |b_{1j} - b_{1k}|^2 \\ &\leq b_{jk}^2 \,, \end{aligned}$$

since the weights b verify the triangle inequality. Under the assumptions of Theorem 8.1 on $\mathcal{H}$, we have, in terms of distribution-based distances:

$$d_{\mathcal{H}}(\mathcal{D}_i, \mathcal{D}_j) \leq |\delta_i - \delta_j| \leq b_{ij} \,.$$

The minimum of each g_α is attained at $x^\alpha = \frac{1}{2} + \delta\alpha$, we thus need to assume that r is of order $\sqrt{d}$, and a rescaling leads to the dependency in r. The dependency in σ^2 for $\sigma^2 > 1$ is obtained by taking $\mathcal{D}_i' = \mathrm{Ber}(1/\sigma^2)\sigma^2\mathcal{D}_i$. In this case, we have a noise amplitude of order σ^2 instead of order 1, and a factor $1/\sigma^2$ appears in the mutual information. □

8.4. The All-for-one algorithm: parallel weighted gradient averagings

After providing lower complexity bounds in Theorem 8.1, we present in this section a naive algorithmic approach based on weighted gradient averagings (WGA), that proves to be sample-optimal. Each agent i keeps n shared local models $x_1^k, \ldots, x_n^k$, where x_j^k estimates $x_j^\star$ at iteration k (the knowledge of x_j^k needs to be shared by all agents). At each iteration k, when a sample ξ_j^k is obtained at agent j, it is used by that agent to compute unbiased estimates of $\nabla f_j(x_i^k)$ for all $i \in [n]$. The iterates of the WGA algorithm write as, where $\lambda_{ij} \geq 0$ are such that $\sum_j \lambda_{ij} = 1$ for all $1 \leq i \leq n$:

$$x_i^{k+1} = x_i^k - \eta \sum_{j=1}^n \lambda_{ij} g_j^k(x_i^k) \,, \tag{8.4}$$

for some step size $\eta > 0$. We call this algorithm that consists in performing n parallel WGA algorithms All-for-one (AFO), since every iteration of each gradient averaging for a given node i requires all the other nodes to compute one stochastic gradient for i. WGA is thus equivalent to training models on the mixture of distributions $(\mathcal{D}_j)_j$ with weights $(\lambda_{ij})_j$ for all i.

Theorem 8.2. *Let $(x_i^k)_{1\leq i\leq n,\, k\geq 0}$ be generated with (8.4), and assume that Assumptions 8.2.1,*

8.2.2 and 8.2.3 (strong version) hold. For any $K \geq 0$ and $1 \leq i \leq n$, and for η as in Equation (8.5),[1]

$$\mathbb{E}\left[f_i(x_i^K) - f_i(x_i^\star)\right] \leq \left(f_i(x_i^{(0)}) - f_i(x_i^\star)\right)e^{-\frac{K}{2\kappa}} + \tilde{\mathcal{O}}\left(\frac{\kappa\sigma^2}{\mu K}\sum_{1\leq j\leq n}\lambda_{ij}^2\right) + \sum_{1\leq j\leq n}\lambda_{ij}\frac{b_{ij}^2}{\mu}\,.$$

Let $\varepsilon > 0$. For a specific choice of $\lambda_{ij} = \frac{\mathbb{1}_{\left\{b_{ij}^2 < \varepsilon/2\right\}}}{\mathcal{N}_i^\varepsilon(2b^2/\mu)}$, WGA (8.4) satisfies $\mathbb{E}\left[f_i(x_i^{K_\varepsilon(i)}) - f_i(x_i^\star)\right] \leq \varepsilon$ for a number of oracle calls of:

$$K_i(\varepsilon) = \tilde{\mathcal{O}}\left(\frac{\kappa\sigma^2}{\mu\varepsilon}\frac{1}{\mathcal{N}_i^\varepsilon(2b^2/\mu)}\right)\,,$$

where $\mathcal{N}_i^\varepsilon$ is previously defined in Theorem 8.1.

Since the oracle complexity of the WGA algorithm matches that of our lower bound, this proves that our lower bound is optimal. However, this algorithm may be difficult to use in practice: *(i)* the choice of λ_{ij} is an explicit function of distribution distances (b_{ij}) (defined in Assumption 8.2.3) that can be (statistically speaking) as hard to compute as solving our optimization problem; and *(ii)* the memory requirements and computation/communication costs of WGA can be prohibitive for large n and large ε (they scale with $\mathcal{N}_i^\varepsilon$ for agent i). Note that the strong version of Assumption 8.2.3 used in Theorem 8.2 can be replaced by a more classical uniform bound of the form $\|\nabla f_i - \nabla f_j\| \leq b_{ij}$.

We first begin by solving this latter issue – an algorithmic one – in the next section, by introducing and studying the All-for-all algorithm. We discuss *(i)* in Section 8.6, where we provide scenarii over which statistical theoretical guarantees can be derived on the error made by estimating these distribution distances using only a few samples.

Proof of Theorem 8.2. We begin by proving the following descent lemma.

Lemma 8.4.1. *Let $F : \mathbb{R}^d \to \mathbb{R}$ be L-smooth and μ-strongly convex and $G : \mathbb{R}^d \to \mathbb{R}$ be differentiable. Consider the iterates generated by:*

$$y^{k+1} = y^k - \eta g^k\,,$$

where $\mathbb{E}\left[g_k|y^k\right] = \nabla G(y^k)$ and $\mathbb{E}\left[\left\|g^k - \nabla G(y^k)\right\|^2|y^k\right] \leq \sigma_g^2$. Then, we have, where $y^\star$ minimizes F, as long as $\eta \leq 1/L$:

$$\mathbb{E}\left[F(y^{k+1}) - F(y^\star)\right] \leq (1-\eta\mu)\mathbb{E}\left[F(y^k) - F(y^\star)\right] + \frac{\eta}{2}\mathbb{E}\left[\left\|\nabla F(y^k) - \nabla G(y^k)\right\|^2\right] + \frac{\eta^2\sigma_g^2 L}{2}$$

Proof Lemma 8.4.1. We use smoothness of F:

$$\mathbb{E}\left[F(y^{k+1}) - F(y^k)\right] \leq -\eta\mathbb{E}\left[\langle g^k, \nabla F(y^k)\rangle\right] + \frac{\eta^2 L}{2}\mathbb{E}\left[\left\|g^k\right\|^2\right]\,.$$

Then, using $\mathbb{E}\left[\left\|g^k\right\|^2\right] \leq \mathbb{E}\left[\left\|\nabla G(y^k)\right\|^2\right] + \sigma_g^2$, and $-\eta\mathbb{E}\left[\langle g^k, \nabla F(y^k)\rangle\right] = -\eta\mathbb{E}\left[\langle \nabla G(y^k), \nabla F(y^k)\rangle\right] = -\frac{\eta}{2}\mathbb{E}\left[\left\|\nabla G(y^k)\right\|^2 + \left\|\nabla F(y^k)^2\right\| - \left\|\nabla G(y^k) - \nabla F(y^k)\right\|^2\right]$, we obtain that:

$$\mathbb{E}\left[F(y^{k+1}) - F(y^k)\right] \leq -\frac{\eta}{2}\mathbb{E}\left[\left\|\nabla F(y^k)\right\|^2\right] - \frac{\eta}{2}\mathbb{E}\left[\left\|\nabla G(y^k)\right\|^2\right](1-\eta L)$$

[1] $\tilde{\mathcal{O}}$ hides logarithmic and constant factors

$$+\frac{\eta}{2}\mathbb{E}\left[\left\|\nabla F(y^k)-\nabla G(y^k)\right\|^2\right]+\frac{\sigma_g^2\eta^2 L}{2}\,.$$

Finally, we conclude using $-\frac{\eta}{2}\mathbb{E}\left[\left\|\nabla F(y^k)\right\|^2\right]\leq\eta\mathbb{E}\left[F(y^k)-F(y^\star)\right]$ and $\eta<1/L$. □

Let $i\in[n]$. To prove Theorem 8.2, we now use Lemma 8.4.1 to study the sequence $y^k=x_i^k$ with $F=f_i$, $G=\sum_j\lambda_{ij}f_j:=f^\lambda$ and $g^k=\sum_j\lambda_{ij}g_j^k(x_i^k)$. For all $x\in\mathbb{R}^d$, using Assumption 8.2.3:

$$\begin{aligned}\left\|\nabla f_i(x)-\nabla f^\lambda(x)\right\|^2&\leq\sum_{j=1}^n\lambda_{ij}\|f_i(x)-f_j(x)\|^2\\&=\sum_{j=1}^n\lambda_{ij}\|\mathbb{E}\left[\nabla_x F(x,\xi_i)\right]-\mathbb{E}\left[\nabla_x F(x,\xi_j)\right]\|^2\\&\leq\sum_{j=1}^n\lambda_{ij}b_{ij}^2\,,\end{aligned}$$

since $\nabla_x F(x,\cdot)\in\mathcal{H}$ and $d_{\mathcal{H}}(\mathcal{D}_i,\mathcal{D}_j)\leq b_{ij}$. Then, using the independence (conditionally on x_i^k) of the $(g_j^k)_j$ and $\mathbb{E}\left[g_j^k(x_i^k)|x_i^k\right]=\nabla f_j(x_i^k)$:

$$\mathbb{E}\left[\left\|\sum_j\lambda_{ij}g_j^k(x_i^k)-\sum_j\lambda_{ij}\nabla f_j(x_j^k)\right\|^2\right]=\sum_j\mathbb{E}\left[\left\|\lambda_{ij}\left(g_j^k-\nabla f_j(x_j^k)\right)\right\|^2\right]\leq\sum_j\lambda_{ij}^2\sigma^2\,.$$

Consequently,

$$\mathbb{E}\left[f_i(x_i^{k+1})-f_i(x_i^\star)\right]\leq(1-\eta\mu)\mathbb{E}\left[f_i(x_i^k)-f_i(x_i^\star)\right]+\frac{\eta}{2}\sum_{j=1}^n\lambda_{ij}b_{ij}^2+\frac{\eta^2\sigma^2 L}{2}\sum_{j=1}^n\lambda_{ij}^2\,.$$

Writing $H^k=(1-\eta\mu)^{-k}\mathbb{E}\left[f_i(x_i^k)-f_i(x_i^\star)\right]$, unrolling the recursion leads to:

$$H^K\leq H^0+\frac{\eta}{2}\sum_{k<K}(1-\eta\mu)^{-k}\sum_{j=1}^n\lambda_{ij}b_{ij}^2+\sum_{k<K}(1-\eta\mu)^{-k}\frac{\eta^2\sigma^2 L}{2}\sum_{j=1}^n\lambda_{ij}^2\,.$$

Finally, using $\sum_{k<K}(1-\eta\mu)^{-k}\leq\frac{(1-\eta\mu)^{-K}}{\eta\mu}$, we have:

$$\mathbb{E}\left[f(x_i^K)-f(x_i^\star)\right]\leq(1-\eta\mu)^K(f(x_i^0)-f(x_i^\star))+\frac{\eta\sigma^2 L}{2\mu}\sum_{j=1}^n\lambda_{ij}^2+\frac{1}{2\mu}\sum_{j=1}^n\lambda_{ij}b_{ij}^2\,.$$

Using $(1-\eta\mu)^K\leq e^{-K\eta\mu}$, we optimize of η. For

$$\eta_i=\min\left\{\frac{1}{2L}\,,\,\frac{1}{\mu K}\ln\left(\frac{2\mu^2K(f(x_i^0)-f(x_i^\star))}{\sigma^2L\sum_j\lambda_{ij}^2}\right)\right\}\,,\tag{8.5}$$

we obtain:

$$\begin{aligned}\mathbb{E}\left[f(x_i^K)-f(x_i^\star)\right]&\leq(f(x_i^0)-f(x_i^\star))e^{-K/\kappa}\\&\quad+\frac{\sigma^2L}{\mu^2K}\ln\left(\frac{2\mu^2K(f(x_i^0)-f(x_i^\star))}{\sigma^2L\sum_j\lambda_{ij}^2}\right)\sum_j\lambda_{ij}^2+\frac{1}{2\mu}\sum_{j=1}^n\lambda_{ij}b_{ij}^2\,,\end{aligned}$$

leading to the first part of Theorem 8.2. For the second part, we simply plug the expression of λ_{ij} in the proven formula. □

8.5. The All-for-all algorithm

Algorithm 8.1: *All-for-all* algorithm

1: Step size $\eta > 0$, matrix $W \in \mathbb{R}^{n\times n}$, initialization $x_1^0 = \ldots = x_n^0 \in \mathbb{R}^d$ (x_i^0 at agent i).
2: **for** $k = 0, 1, 2, \ldots K-1$ **do**
3: Agents $1 \le j \le n$ compute $g_j^k(x_j^k)$ and broadcast it to all agents i such that $W_{ij} > 0$.
4: For $i = 1, \ldots, n$, update:

$$x_i^{k+1} = x_i^k - \eta \sum_{j\,:\,W_{ij}>0} W_{ij} g_j^k(x_j^k)\,.$$

5: **end for** Return x_i^K for agent i

In this section, we present the All-for-all algorithm (AFA), an adaptation of the weighted gradient averaging algorithm. For $1 \le i \le n$, initialize $x_i^0 = x_0 \in \mathbb{R}^d$. At iteration k, let $x_i^k \in \mathbb{R}^d$ be agent i's current estimate of $x_i^\star$, and denote $x^k = (x_i^k)_{1\le i\le n} \in \mathbb{R}^{n\times d}$. For a step size $\eta > 0$ and a matrix $W \in \mathbb{R}^{n\times n}$ with non-negative entries (remarkably and as discussed later, W will not necessarily verify $\sum_j W_{ij} = 1$), iterates of the *all-for-all* algorithm are generated with Algorithm 8.1. In Theorem 8.3, we control the averaged local generalization error amongst all agents:

$$F^k = \frac{1}{n}\sum_{i=1}^n f_i(x_i^k) - f_i(x_i^\star)\,, \quad k \ge 0\,.$$

Theorem 8.3 (All-for-all algorithm)**.** *Let $K > 0$, $\eta > 0$, and W a matrix of the form $W = \Lambda\Lambda^\top$ for some stochastic matrix $\Lambda = (\lambda_{ij})_{1\le i,j\le n}$. Assume that Assumptions 8.2.1, 8.2.2 and 8.2.3 (weak version) hold. For a certain choice of η detailed in the proof, the iterates $(x_i^k)_{k\ge 0, 1\le i\le n}$ generated with Algorithm 8.1 verify:*

$$\frac{1}{K}\sum_{k<K}\mathbb{E}\left[F^k\right] \le \frac{LB^2}{K} + 2\sqrt{\frac{B^2\sigma^2}{NK}\sum_{1\le i,j\le n}\lambda_{ij}^2} + \frac{1}{n}\sum_{1\le i,j\le n}\lambda_{ij}\left(f_j(x_i^\star) - f_i(x_i^\star)\right)\,,$$

where $B^2 \ge \frac{1}{n}\sum_{i=1}^n \left\|x_i^{(0)} - x_i^\star\right\|^2$ and if $\mu > 0$:

$$\mathbb{E}\left[F^K\right] \le F^0 e^{-\frac{K}{2\kappa}} + \tilde{\mathcal{O}}\left(\frac{\kappa\sigma^2}{K\mu n}\sum_{1\le i,j\le n}\lambda_{ij}^2\right) + \frac{1}{n}\sum_{1\le i,j\le n}\lambda_{ij}\left(f_j(x_i^\star) - f_i(x_i^\star)\right)\,.$$

As for the AFO algorithm, we can deduce from this result the number of oracle calls required by the All-for-all algorithm to reach an averaged ε-generalization, under the idealistic setting where the distribution-based distances b_{ij} are accessible.

Corollary 8.5.1. *Let $\varepsilon > 0$. Under the same assumptions as in Theorem 8.3, for a choice of matrix $W = \Lambda\Lambda^\top$ where $\lambda_{ij} = \frac{\mathbb{1}_{\{b_{ij}^2/\mu < \varepsilon\}}}{\mathcal{N}_i^\varepsilon(b^2/\mu)}$, the* ALL-FOR-ALL *algorithm (Algorithm 8.1) returns $(x_i^K)_{1\leq i\leq n}$ satisfying $\frac{1}{n}\sum_{i=1}^n f_i(x_i^{K_\varepsilon}) - f_i(x_i^\star) \leq \varepsilon$, for a number of oracle calls satisfying:*

$$K_\varepsilon \leq 2\max\left(\frac{\kappa\sigma^2}{\varepsilon\mu}\frac{1}{n}\sum_{i=1}^n \frac{1}{\mathcal{N}_i^\varepsilon(b^2/\mu)}, \kappa\right)\ln\left(\varepsilon^{-1}F^0\right).$$

Denoting $K_\varepsilon(i)$ the oracle complexity of the WGA algorithm - that matches the lower bound -, we observe that the ALL-FOR-ALL algorithm reaches an averaged ε-generalization with an number of oracle calls K_ε, of $K_\varepsilon \leq \frac{1}{n}\sum_i K_\varepsilon(i)$. The speedup in comparison with a no-collaboration strategy (all agents locally performing SGD) is $\frac{1}{n}\sum_i \frac{1}{\mathcal{N}_i^\varepsilon(b^2/\mu)}$: the mean of all local speedups.

Remark 8.5.1. *In Theorem 8.3, as its proof shows, the quantities $b_{ij}^2/(2\mu)$ in the last term can in fact be replaced by the quantities $f_i(x_j^\star) - f_i(x_i^\star)$, that control how well the optimal model for j generalizes for i, and the bias induced by* ALL-FOR-ALL *iterations is a weighted average of these quantities. Note that in our lower bound (Theorem 8.1), we enforce that the functions considered are required to satisfy $f_i(x_j^\star) - f_i(x_i^\star) \leq b_{ij}^2/(2\mu)$. We believe this notion of function proximity that we leverage to be the weakest achievable in our setting; no prior work uses such a mild proximity assumption.*

Perhaps surprisingly, matrix W is in general *not* a gossip matrix (*i.e.* such that $W\mathbb{1} = \mathbb{1}$): agent i does not aggregate a convex combination of stochastic gradients, but a combination with scalars that do not necessarily sum to 1. We thus cannot say that the ALL-FOR-ALL algorithm acts as if, in parallel, each agent i trains a model on the mixture of distributions $\mathcal{D}_j$ with weights W_{ij}. In fact, as the analysis shows below, agent i trains a model on the mixture of distributions, with weights λ_{ij}, if Λ is a stochastic square root of matrix W ($\Lambda\Lambda^\top = W$), as in the AFO algorithm. In order to account for inter-dependencies between agents that do not directly share information, the *all-for-all gradient filtering* uses weights W_{ij} to aggregate information, instead of λ_{ij}. Propagating information using a matrix W, that induces a similarity graph G_W on $\{1,\dots,n\}$, such that $(ij) \in E_W$ if $W_{ij} > 0$, is quite natural [VBT17, BGTT18]; yet, ours is the first analysis to give such precise generalization error bounds, through the use of a stochastic optimization framework.

In comparison to Theorem 8.3, the classical personalized FL approaches that consider personalized local models of the form $x_i = \bar{x} - \delta_i$, where $\bar{x}$ is some global quantity shared by all agents, perturbed (and personalized) by some local quantity δ_i (*e.g.* averaging between local and a global models), can be seen as the special instances where, for all i, we have $\lambda_{ii} = 1-\alpha_i$ and $\lambda_{ij} = \frac{\alpha_i}{n-1}$ if $i \neq j$ for some α_i, and leads to bias terms of the form $\frac{1}{n}\sum_i \frac{\alpha_i}{N-1}\sum_{j\neq i} b_{ij}$ [DKM20, MMRS20, FMO20]. Full and naive collaboration (a single model trained for all users) corresponds to $\lambda_{ij} = 1/n$ for all i,j, and leads to a bias term of $\frac{1}{N^2}\sum_{i,j} b_{ij}$. The degrees of freedom offered by our matrix W (and by coefficients λ_{ij}) enable pairwise agent adaptation, and tighter generalization guarantees and bias/variance tradeoffs.

Proof of Theorem 8.3. We recall that for a stochastic matrix Λ, we defined

$$f^\Lambda(y) = \frac{1}{n}\sum_{i=1}^n f_i\Big(\sum_{j=1}^n \lambda_{ij} y_j\Big), \quad y = (y_1,\dots,y_n) \in \mathbb{R}^{n\times d}.$$

Then, y^Λ is defined as a minimizer of f^Λ, and we write $x^\star = (x_1^\star,\dots,x_n^\star)$ where $x_i^\star$ is the minimizer of f_i.

We first begin with the following simple lemmas.

Lemma 8.5.1. *If Assumption 8.2.3 (weak version) holds, then for all $i, j = 1, \dots, n$, we have:*

$$f_i(x_j^\star) - f_i(x_i^\star) \leq \frac{b_{ij}^2}{2\mu}\,.$$

Proof. Using strong-convexity of f_i and $\nabla f_i(x_i^\star)$:

$$\begin{aligned} f_i(x_j^\star) - f_i(x_i^\star) &\leq \frac{1}{2\mu}\left\|\nabla f_i(x_j^\star)\right\|^2 \\ &= \frac{1}{2\mu}\left\|\nabla f_i(x_j^\star) - \nabla f_i(x_i^\star)\right\|^2 \\ &= \frac{1}{2\mu}\left\|\mathbb{E}\left[\nabla_x F(x_j^\star, \xi_i)\right] - \mathbb{E}\left[\nabla_x F(x_j^\star, \xi_i)\right]\right\|^2 \\ &\leq \frac{b_{ij}^2}{2\mu}\,, \end{aligned}$$

where the last inequality is deduced using the weak version of Assumption 8.2.3. □

Lemma 8.5.2. *If Λ is a stochastic matrix,*

$$f^\Lambda(y^\Lambda) - \bar{f}(x^\star) \leq \frac{1}{n}\sum_{1\leq i,j\leq n} \lambda_{ij}\big(f_i(x_j^\star) - f_i(x_i^\star)\big)\,.$$

Proof. Writing the optimality of y^Λ gives:

$$\begin{aligned} f^\Lambda(y^\Lambda) &\leq f^\Lambda(x^\star) \\ &= \frac{1}{n}\sum_i f_i(\sum_j \lambda_{ij} x_j^\star) \\ &\leq \frac{1}{n}\sum_{1\leq i,j\leq n} \lambda_{ij} f_i(x_j^\star)\,, \end{aligned}$$

where we used convexity of each f_i. Then, subtracting $\bar{f}(x^\star)$ and using stochasticity of Λ:

$$f^\Lambda(y^\Lambda) - \bar{f}(x^\star) \leq \frac{1}{n}\sum_{1\leq i,j\leq n} \lambda_{ij}(f_i(x_j^\star) - f_i(x_i^\star))\,.$$

□

We are now armed to prove Theorem 8.3. Recall that $x^k \in \mathbb{R}^{d\times n}$ is defined through the recursion:

$$x^{k+1} = x - k - \eta W G^k(x^k)\,.$$

We define the virtual iteration $y^k \in \mathbb{R}^{d\times n}$ as:

$$y^0 = x^0 \quad , \qquad y^{k+1} = y^k - \nabla f^\Lambda(y^k)\,.$$

Since $\nabla f^\Lambda(y^k) = \Lambda^\top \nabla \bar{f}(\lambda y^k)$, we have that:

$$y^0 = x^0 \quad , \qquad y^{k+1} = y^k - \Lambda^\top \nabla \bar{f}(\Lambda y^k)\,,$$

and, multiplying by Λ:

$$\Lambda y^0 = \Lambda x^0 \quad , \qquad \Lambda y^{k+1} = \Lambda y^k - \Lambda\Lambda^\top \nabla \bar{f}(\Lambda y^k)\,.$$

Since $x_i^0 = x_j^0$ for all i, j, we have that $\Lambda x^0 = x^0 = y^0 = \Lambda y^0$. Then, since by definition we have that $\Lambda\Lambda^\top = W$, we notice that the iterates $\tilde{x}^k = \Lambda y^k$ satisfy:

$$\tilde{x}^0 = x^0 \quad , \qquad \tilde{x}^{k+1} = \tilde{x}^k - W\nabla \bar{f}(\tilde{x}^k)\,.$$

This is the same recursion as satisfied by x^k with the same initialization: we can conclude that $\tilde{x}^k = x^k$ for all k. We have the following bias-variance decomposition, where the inequality is a consequence of Lemma 8.5.2:

$$\begin{aligned} F^k &= \bar{f}(x^k) - \bar{f}(x^\star) \\ &= f^\Lambda(y^k) - f^\Lambda(y^\Lambda) + f^\Lambda(y^\Lambda) - \bar{f}(x^\star) \\ &\leq f^\Lambda(y^k) - f^\Lambda(y^\Lambda) + \frac{1}{n}\sum_{1\leq i,j\leq n} \lambda_{ij}\frac{b_{ij}^2}{2\mu}\,. \end{aligned}$$

We thus need to upper-bound the optimization term $f^\Lambda(y^k) - f^\Lambda(y^\Lambda)$. We recall that y^k verifies the recursion:

$$y^{k+1} = y^k - \eta\nabla G_\Lambda^k(y^k)\,,$$

for

$$G_\Lambda^k(y) = \frac{1}{n}\Big(\sum_{i=1}^n \lambda_{ij} g_i^k((\Lambda y^k)_i)\Big)_{1\leq j\leq n}\,,$$

that verifies:

$$\begin{aligned} &\mathbb{E}\left[G_\Lambda^k(y)\right] = \nabla f^\Lambda(y)\,, \\ &\mathbb{E}\left[\left\|G_\Lambda^k(y) - \nabla f^\lambda(y)\right\|^2\right] \leq \frac{\sigma^2}{N^2}\sum_{1\leq i,j\leq n}\lambda_{ij}^2\,. \end{aligned}$$

The function f^Λ is however not necessarily strongly convex. However, since $\nabla^2 f^\Lambda(y) = \Lambda^\top\nabla^2\bar{f}(\Lambda y)\Lambda$ and $\bar{f}$ is L/n-smooth and μ/n-strongly convex, f^Λ is L/n-relatively smooth and μ/n-relatively strongly convex [BBT17] with respect to $\frac{1}{2}\|y\|_W^2 = \frac{1}{2}y^\top W y$. Note also that the spectral radius of W is 1, since Λ is stochastic. For the storngly convex case, we use Lemma 8.B.1 that we prove in the appendix of this chapter, while for the convex case we can use classical SGD analysis with our noise and smoothness parameters [GG23, Theorem 5.3]. This leads to, if $\mu > 0$:

$$\begin{aligned} \mathbb{E}\left[f^\Lambda(y^K) - f^\Lambda(y^\Lambda)\right] \leq\ & (f^\Lambda(y^0) - f^\Lambda(y^\Lambda))e^{-\frac{K}{2\kappa}} \\ &+ \frac{\kappa\sigma^2}{K\mu n}\ln\left(\frac{2\mu^2 K(f^\Lambda(y^0) - f^\Lambda(y^\Lambda))}{\sigma^2 L\sum_{i,j}\frac{1}{n}\lambda_{ij}^2}\right)\sum_{1\leq i,j\leq n}\lambda_{ij}^2\,, \end{aligned}$$

for a choice of stepsizes of:

$$\eta = \min\left\{\frac{1}{2L}\,,\ \frac{1}{\mu K}\ln\left(\frac{2\mu^2 K(f^\Lambda(y^0) - f^\Lambda(y^\Lambda))}{\sigma^2 L\sum_{i,j}\frac{1}{n}\lambda_{ij}^2}\right)\right\}, \tag{8.6}$$

while if $\mu = 0$:

$$\mathbb{E}\left[f^\Lambda\Big(\frac{1}{K}\sum_{k<K} y^k\Big) - f^\Lambda(y^\Lambda)\right] \leq \frac{L}{K}\frac{1}{n}\sum_{i=1}^n \left\|x_i^{(0)} - x_i^\star\right\|^2 + 2\sqrt{\frac{1}{NK}\sum_{i=1}^n\left\|x_i^{(0)} - x_i^\star\right\|^2\frac{\sigma^2}{n}\sum_{1\leq i,j\leq n}\lambda_{ij}^2}$$

for $\gamma = \min\left(1/(2L), \sqrt{\frac{\frac{L}{n}\sum_{i=1}^n \left\|x_i^{(0)} - x_i^\star\right\|^2}{K\frac{\sigma^2}{n}\sum_{1\leq i,j\leq n}\lambda_{ij}^2}}\right)$ concluding the proof. □

After providing the optimization tools and results to answer for the shortcomings of weighted gradient averagings, we now turn to quantifying the impact of the use of estimated values $\hat{b}_{ij}$ of b_{ij}, in order to close the loop.

8.6. Estimation of $d_{\mathcal{H}}(\mathcal{D}_i, \mathcal{D}_j)$ as a pre-processing step

The sample complexity of estimating the distances $d_{\mathcal{H}}(\mathcal{D}_i, \mathcal{D}_j)$ depends on the complexity of the function space $\mathcal{H}$. While the estimation of Wasserstein or total variation distances are usually hard (in $\mathcal{O}(1/S^{1/d})$ where d is the ambient dimension and S the number of samples available for the estimation, see *e.g.* [WB19]), maximum mean discrepancy (MMD) distances often exhibit lower sample complexities in $O(1/\sqrt{S})$ [SGF+10]. Moreover, explicit assumptions on the loss function can also provide low sample complexities, as shown below for quadratic loss functions. Yet, the results presented in this section can be generalized beyond linear models with squared losses, as long as concentration inequalities for controlling how far empirical distributions are from the true distribution in terms of distance $d_{\mathcal{H}}$ are known.

In order to formulate statistical results for the estimation of the pairwise distribution-based distances $d_{\mathcal{H}}(\mathcal{D}_i, \mathcal{D}_j)$, we need to make additional structural assumptions, on both $\mathcal{H}$ and the distributions. Inspired by [CHMS21], we focus on analyzing an instance of our general GEM setting for quadratic losses and linear models, under which the generalization error of a given agent i writes as:

$$f_i(x) = \frac{1}{2}\mathbb{E}\left[\left(a_i^\top x - b_i\right)^2\right], \quad x \in \mathbb{R}^d,$$

where $z_i = (a_i, b_i)$ is a random variable on $\mathbb{R}^d \times \mathbb{R}$. The stochastic gradients thus write as $\nabla_x F(x, \xi_i) = (a_i^\top x - b_i)a_i$ for $z_i = (a_i, b_i)$, and are thus linear functions of $\xi_i = z_i z_i^\top$. Hence, Assumption 8.2.3 (weak version) is satisfied for $\mathcal{H}$ the set of $D^\star$-Lipschitz and affine functions, where $D^\star$ bounds all $\|x_i^\star\|$ for $1 \leq i \leq n$, leading to:

$$d_{\mathcal{H}}(\mathcal{D}_i, \mathcal{D}_j) \leq D^\star \|\mathbb{E}\left[\xi_i\right] - \mathbb{E}\left[\xi_j\right]\|.$$

We make the following assumption on the law $\mathcal{D}_i$ of the random variables ξ_i: they are non-isotropic subgaussian random variables, that thus benefit from concentration inequalities that are dimension-independent [EM21, KL17].

Assumption 8.6.1. *For some non-negative symetric matrix Σ and all $1 \leq i \leq n$, ξ_i are centered and Σ-subgaussian:*

$$\mathbb{P}\left(\xi_i^\top y \geq u\right) \leq \exp\left(-\frac{u^2}{2y^\top \Sigma y}\right), \quad \forall y \in \mathbb{R}^{(d+1)^2}, \quad \forall u > 0,$$

and we denote as ν^2 the largest eigenvalue of Σ, and $d_{\text{eff}} = \frac{\|\Sigma\|_2}{\nu^2}$ its effective dimension.

Importantly, note that d_{eff} can be arbitrarily smaller than the ambient dimension - for the MNIST dataset, d_{eff} is less than 3, while the ambient dimension is 712 [EM21]. Depending on a smaller dimension is also an assumption that [CHMS21] use in their work by exploiting shared representations.

We now formulate a structural assumption on the set of agents: there are M clusters $\mathcal{C}_1, \ldots, \mathcal{C}_M$ of C agents each (to ease notations, with a total number of agents $n = MC$).

Within each cluster, agents distributions share the same objective, and clusters are "well-separated". These models are popular for modelling population heterogeneity and provide a formal framework for clustering problems; we refer the interested reader to [MM10] for a detailed survey on the subject.

Assumption 8.6.2 (Well-separated clusters of agents)**.** *For $M, C \geq 1$, n writes as $n = MC$ and there exists $\{\mathcal{C}_1, \dots, \mathcal{C}_M\}$ a partition of $\{1, \dots, n\}$, $\mu_1, \dots, \mu_m$ such that for all $1 \leq m \leq M$, $|\mathcal{C}_m| = C$ and for all $i, j \in \mathcal{C}_m$, we have $\mathbb{E}[\xi_i] = \mathbb{E}[\xi_j] = \mu_m$. We denote $\Delta^2 = \min_{m \neq m'} \|\mu_m - \mu_{m'}\|^2$ and assume that $\Delta^2 > 0$.*

When distribution-based distances were given (as in Corollary 8.5.1), Algorithm 8.1 achieved the optimal collaboration speedup, linear in $1/C$ under Assumption 8.6.2 and for small enough target precision ε. The cluster model is thus the natural baseline for our problem. In the case where agents estimate with whom to collaborate as we do in the sequel, reaching this collaboration speedup of $1/C$ will hence prove the effectiveness of the approach.

We assume that agents possess a limited number of samples. More precisely, for $1 \leq i \leq n$ and $S, K \geq 1$, agent i possesses $K+S$ *i.i.d.* samples of drawn from $\mathcal{D}_i$, S of which are dedicated to estimating who to collaborate with, the K remaining dedicated to the optimization process *i.e.* to running All-for-all iterations for a number K of oracle calls.

For $1 \leq i \leq n$, let $\hat{\mu}_i$ be an estimation of $\mathbb{E}[\xi_i]$ made with S *i.i.d.* samples $\xi_{i,1}, \dots, \xi_{i,S}$, and for $1 \leq i, j \leq n$ let $\hat{b}_{ij}$ be the following estimation of $d_{\mathcal{H}}(\mathcal{D}_i, \mathcal{D}_j)$:

$$\hat{\mu}_i = \frac{1}{S} \sum_{s=1}^{S} \xi_{i,s}\,, \quad \hat{b}_{ij} = \|\hat{\mu}_i - \hat{\mu}_j\|\,.$$

Computing these distances can be done using only $\tilde{\mathcal{O}}(N)$ communications (rather than the n^2 communications of a naive approach) by performing randomized gossip communications [BGPS06] on the complete graph.

Theorem 8.4 (All-for-all with estimated biases)**.** *Assume that Assumptions 8.2.1, 8.2.2, 8.2.3 (for some unknown biases b_{ij}), 8.6.1 and 8.6.2 hold. Under the setting described, let $\hat{\Lambda}$ be the stochastic matrix with entries*

$$\hat{\lambda}_{ij} = \frac{\mathbb{1}_{\{\hat{b}_{ij}^2 \leq u\}}}{\sum_{F=1}^{n} \mathbb{1}_{\{\hat{b}_{iF}^2 \leq u\}}}\,,$$

for some $u > 0$ that verifies $u \geq \frac{4\nu^2 d_{\text{eff}}}{S}$. The All-for-all algorithm with $W = \hat{\Lambda}\hat{\Lambda}^\top$ outputs $(x_i^K)_{1 \leq i \leq n}$ verifying, where $b_{\max} = \max_{i,j} b_{ij}$:

$$\mathbb{E}\left[F^K\right] \leq F^0 e^{-\frac{K}{2\kappa}} + \tilde{\mathcal{O}}\left(\frac{\kappa\sigma^2}{K\mu}\left(\frac{1}{C} + Ce^{-\frac{Su}{8\nu^2}}\right)\right) + \frac{2D^{\star 2} b_{\max} e^{-\frac{S\max(\Delta^2 - 2u, 2u^2)}{8\nu^2}} + 4u^2 \mathbb{1}_{\{2u \geq \Delta\}}}{\mu}\,,$$

where the mean is taken over both biases estimates $(\hat{b}_{ij})$ and gradient estimates (g_i^k).

Corollary 8.6.1. *Under the same assumptions as Theorem 8.4 and for $\varepsilon > 0$, the* All-for-all *algorithm with estimated biases as described above reaches an averaged generalization error of ε as long as:*

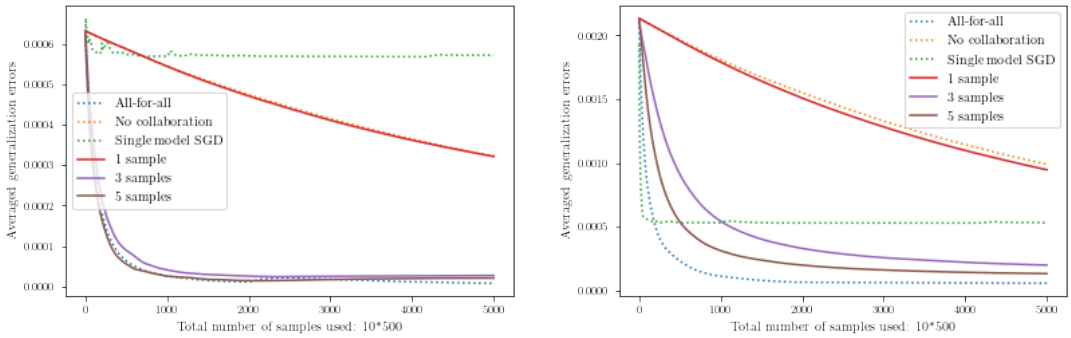
$$S = \tilde{\Omega}\Big(\frac{\nu^2 d_{\text{eff}}}{\Delta^2}\Big)\,, \quad C = \tilde{\Omega}\Big(\frac{\nu^2 d_{\text{eff}}}{\varepsilon}\Big)\,, \quad KC = \tilde{\Omega}\Big(\frac{\kappa\sigma^2}{\varepsilon\mu}\Big)\,, \quad K = \tilde{\Omega}(\kappa)\,.$$

Forgetting about the logarithmic factors, only a bounded number of local samples for each user (S and K) are required to reach an averaged arbitrarily small generalization error

$\varepsilon > 0$, in the limit with an arbitrary large number of agents (n and C). Indeed, due to our regularity assumptions, K – the number of samples kept for the optimization problem – is required only to be of order κ, the condition number of the problem. The number of samples S used for estimating the biases is required to be of order $\nu_1^2 d_{\text{eff}}/\Delta^2$, the "signal-to-noise" ratio of our mixture model [MM10, SSR06], a natural quantity to depend on. Corollary 8.6.1 hence shows that the optimal collaboration speedup is achieved, up to logarithmic factors: in order to reach an arbitrary small generalization error $\varepsilon > 0$, are only required constant orders for S and K (the number of samples locally available) if the number of agents is large enough *i.e.* if $n = \tilde{\Omega}(M/\varepsilon)$, where M is the number of clusters *i.e.* we have a linear speedup in the clusters population. We numerically illustrate our theory in Appendix 8.7 on synthetic datasets, with clustered agents (as in this section), as well as in a setting where agents are distributed according to a more general "distribution of agent".

A closely related work [GCYR20] also studies a model where agents verify a cluster structure as described in Assumption 8.6.2 for quadratic losses and linear models. Yet, we highlight several differences between their approach and ours. First, [GCYR20] perform an *online* clustering of the agents, as opposed to our pre-training hierarchical approach. While the results we obtain in Theorem 8.4 and Corollary 8.6.1 and those of [GCYR20] have the same linear speedup in the number of agents, ours require no initialization condition. Finally, our algorithm is decentralized, thus leading to improved scalability (especially in terms of the number of clusters) and privacy [CEBM22], if of interest. Finally, not being restricted to clusters in the analysis of the All-for-all algorithm leads to a better collaboration speedup and fairness (in the sense that performance does not impact a few agents) in a non-clustered scenario, where an approach based on clusters would be highly non-optimal for agents that are at the border of the inferred clusters.

8.7. Numerical illustration of our theory



(a) – *Agents in unknown clusters* (b) – *Agents* i.i.d. *distributed*

Figure 8.1 – **All-for-all algorithm in practice**

To test the robustness of our theory, we build toy problems from synthetic datasets, placing ourselves in the scenario we considered throughout this chapter: a large number of agents with heterogeneous data, that each have too few samples available from their local data distribution in order to reach a small generalization error on their own.

In Figure 8.1, we consider $n = 500$ agents, and a quadratic loss function $F(x, \xi = (a, b)) = \frac{1}{2}(a^\top x - b)^2$, for $x, a \in \mathbb{R}^d$ ($d = 100$) and $\mu \in \mathbb{R}^d$. For $i = 1, \dots, 500$, the distribution $\mathcal{D}_i$ of $\xi_i = (a_i a_i^\top, a_i b_i)$ as a centered Gaussian random variable of covariance matrix Σ_i for a_i, and b_i is the sign of $a_i^\top u$ for some fixed $u \in \mathbb{R}^d$, flipped with probability 0.2. In both figures, each agents have 10 samples available for the optimization phase ($K = 10$ oracle calls), corresponding to a total number of samples used of $n \times K = 5000$. We computed and showed

the 500 steps of all ten oracle calls, each step corresponding to the use of the stochastic gradient of a single agent.

The dotted lines represent our baselines. The blue one is the All-for-all algorithm with matrix W exactly as in Corollary 8.5.1 with $b_{ij} = \|\Sigma_i - \Sigma_j\|$. The orange dotted line consists in the no-collaboration baseline: each agents performs SGD on its own without sharing information (corresponds to $W = I_n$). The green dotted line corresponds to the "single-model" approach without personalization: one model is trained for all agents, using SGD and all samples from all agents (corresponding to $W = \frac{1}{n}\mathbb{1}\mathbb{1}^\top$). The choice of the algorithm for the single model approach without collaboration is in fact unimportant, since all algorithms would reach the same asymptotic bias here. The full lines (red, violet and brown) correspond to estimating the pairwise distances from empirical distributions (as in Section 8.6), using respectively $S = 1$, $S = 3$ and $S = 5$ samples.

In Figure 8.1a, we consider $M = 10$ (unkonwn) clusters $\mathcal{C}_1, \dots, \mathcal{C}_M$. All $i \in \mathcal{C}_m$ have the same covariance matrix Σ_m, equal to $I_d/\sqrt{d} + e_m e_m^\top$, where e_m is the m-th element of the canonical basis of $\mathbb{R}^d$. In Figure 8.1b, $\Sigma_i = \mathrm{Diag}(u_1^{(i)}, \dots, u_1^{(d)})/\sqrt{d}$ where the $(u_F^{(i)})$ are *i.i.d.* uniformly distributed in $[0, 1]$. Performing rough estimations of the pairwise distance between agents' local distributions thus appears to be quite robust in both our settings. In the "cluster" setting, this was predicted by our theory, and the numerical results are compelling. In the "*i.i.d.*" setting, using very few samples for the estimation also appears to be very efficient.

Conclusion

In this chapter, we quantified in terms of function and distribution biases, stochastic gradient noise, target precision $\varepsilon > 0$ and functions regularity parameters, the benefit of collaboration between agents for shared minimization using stochastic gradient algorithms. Our lower bound (Theorem 8.1) states that, under prior knowledge on the distances between local distributions, the collaborative speedup can be linear only in the first phase of the optimization when the generalization error is large compared to the distances between distributions. More specifically, for a given agent i, the collaboration speedup is linear in the number of agents that are ε-close to i. Moreover, we show that the All-for-one algorithm allows such a speedup and is thus sample optimal. However, this algorithm requires high computation and communication capacities, a drawback that can be mitigated by the use of a novel algorithm called All-for-all, that benefits from the same collaboration speedup while being cheaper to deploy. Finally, we studied the impact of estimating distances between distributions as a pre-processing step to the optimization phase; under a mixture model assumptions on the agents, we obtain an optimal collaboration speedup.

Appendix of Chapter 8

8.A. Proof of Theorem 8.4

We first start by recalling that, for a Σ-subgaussian random variable ξ, using Theorem 1 from [HKZ12], we have for any $t \geq 0$:

$$\mathbb{P}\left(\|\xi\|^2 \geq d_{\text{eff}}\nu^2 + 2\sqrt{\|\Sigma\|_2 t} + 2\nu^2 t \right) \leq e^{-t}\,.$$

Consequently, for $u \geq \max(\nu^2 d_{\text{eff}}, \|\Sigma\|_2^2/\nu^2)$, we have:

$$\mathbb{P}\left(\|\xi\|^2 \geq 4u \right) \leq e^{-\frac{u}{2\nu^2}}\,.$$

Remarking that $\frac{\|\Sigma\|_2^2}{\nu^2} = \nu^2 \sum_k \frac{\nu_k^4}{\nu^4} \leq \nu^2 \sum_k \frac{\nu_k^2}{\nu^2} = \nu^2 d_{\text{eff}}$, where ν_k^2 are the eigenvalues of Σ, this condition on u is in fact $u \geq \nu^2 d_{\text{eff}}$.

Proof of our Theorem. From Theorem 8.3, we have, conditionally on the S samples used in estimating biases:

$$\mathbb{E}\left[F^K|\hat{\Lambda}\right] \leq F^0 e^{-\frac{K}{2\kappa}} + \tilde{\mathcal{O}}\left(\frac{\kappa\sigma^2}{K\mu n} \sum_{1\leq i,j\leq n} \hat{\lambda}_{ij}^2 \right) + \frac{1}{n} \sum_{1\leq i,j\leq n} \hat{\lambda}_{ij} \frac{b_{ij}^2}{2\mu}\,.$$

We hence need to bound $\mathbb{E}\left[\sum_{i,j} \hat{\lambda}_{ij}^2\right]$ and $\mathbb{E}\left[\sum_{i,j} \hat{\lambda}_{ij} b_{ij}^2\right]$, and start by assuming that $u \geq \frac{4}{S}\nu^2 d_{\text{eff}}$.

First, denoting $b_{\max} = \max_{i,j} b_{ij}$, we have:

$$\begin{aligned}
\mathbb{E}\left[\sum_{i,j} \hat{\lambda}_{ij} b_{ij}^2\right] &= \sum_{i,j} \mathbb{E}\left[\hat{\lambda}_{ij} b_{ij}^2\right] \\
&= \sum_{i,j:\, b_{ij}^2 > 4u} \mathbb{E}\left[\hat{\lambda}_{ij} b_{ij}^2\right] + \sum_{i,j:\, b_{ij}^2 \leq 4u} \mathbb{E}\left[\hat{\lambda}_{ij} b_{ij}^2\right] \\
&\leq b_{\max}^2 \mathbb{P}\left(\hat{b}_{ij}^2 \leq u | b_{ij}^2 > 4u\right) + 4u^2 \mathbb{1}_{\{4u^2 \geq \Delta\}}\,.
\end{aligned}$$

Using a triangle inequality, that gives us $2\|\hat{\mu}_i - \hat{\mu}_j - \mathbb{E}\left[\hat{\mu}_i - \hat{\mu}_j\right]\|^2 \geq b_{ij}^2 - 2\hat{b}_{ij}^2$, $\mathbb{P}\left(\hat{b}_{ij}^2 \leq u | b_{ij}^2 > 4u\right) \leq \mathbb{P}\left(\|\hat{\mu}_i - \hat{\mu}_j - \mathbb{E}\left[\hat{\mu}_i - \hat{\mu}_j\right]\|^2 \geq \frac{b_{ij}^2 - 2u}{2}\right)$. Then, since each ξ_i and ξ_j are Σ-subgaussian (and independent), $\hat{\mu}_i - \hat{\mu}_j - \mathbb{E}\left[\hat{\mu}_i - \hat{\mu}_j\right]$ is $4\Sigma/S$ subgaussian[2], so that using our assumption on

[2] indeed, using the subgaussian norm $\|\cdot\|_{\psi_2}$, for real-valued independent random variables $X_1, \ldots, X_S$ and any $\beta > 0$, $\mathbb{E}\left[e^{\beta \frac{1}{S}\sum_{s=1}^S X_s}\right] = \prod_s \mathbb{E}\left[e^{\frac{\beta}{S} X_s}\right] \leq \prod_s \mathbb{E}\left[e^{C_{\psi_2} \frac{\beta^2}{S^2} \|X_s\|_{\psi_2}}\right] = \mathbb{E}\left[e^{C_{\psi_2} \frac{\beta^2}{S^2} \sum_{s=1}^S \|X_s\|_{\psi_2}}\right]$, so that

u, we can use Theorem 1 of [HKZ12]:

$$\mathbb{P}\left(\hat{b}_{ij}^2 \leq u | b_{ij}^2 > 4u\right) = \mathbb{P}\left(\hat{b}_{ij}^2 \leq u | b_{ij}^2 > \max(4u, \Delta^2)\right) \leq 2e^{-\frac{S\max(2u,\Delta^2-2u)}{8\nu^2}}.$$

Then, for $1 \leq i \leq n$, let $\hat{\mathcal{N}}_i = \sum_{j=1}^n \mathbb{1}_{\{\hat{b}_{ij} \leq u\}}$. Fix $1 \leq m \leq M$. We have:

$$\begin{aligned}\mathbb{P}\left(\forall i \in \mathcal{C}_m, \|\hat{\mu}_i - \mu_m\|^2 \leq u\right) &= 1 - \mathbb{P}\left(\exists i \in \mathcal{C}_m, \|\hat{\mu}_i - \mu_m\|^2 > u\right) \\ &\geq 1 - 2Ce^{-\frac{Su}{8\nu^2}},\end{aligned}$$

so that $\mathbb{E}\left[\sum_{i\in\mathcal{C}_M}\sum_{1\leq j\leq n}\lambda_{ij}^2\right] = \mathbb{E}\left[\frac{1}{C}\sum_{i\in\mathcal{C}_m}\frac{1}{\hat{\mathcal{N}}_i}\right] \leq \frac{1}{C} + 2Ce^{-\frac{Su}{8\nu^2}}$, concluding the proof. We then prove the resulting corollary by taking $u = \Delta/4$, and the condition on u translates into $S \geq 16\frac{\nu^2 d_{\text{eff}}}{\Delta^2}$.

□

8.B. SGD under strong-convexity and smoothness assumptions

Lemma 8.B.1 (SGD, s.c. and smooth)**.** *Define $\|x\|_A^2 = x^\top Ax$ for some non-negative and symmetric matrix A. Let $f : \mathcal{X} \to \mathbb{R}$ μ-relatively strongly convex and L-relatively smooth with respect to $\frac{1}{2}\|x\|_A^2$. Let $(f_t, g_t)_{t\geq 0}$ be first order oracle calls such that for all $t \geq 0$:*

$$\forall x \in \mathcal{X}\,, \quad \begin{cases}\mathbb{E}\left[f_t(x)\right] = f(x)\,,\\ \mathbb{E}\left[g_t(x)\right] = \nabla f(x)\,,\\ \mathbb{E}\left[\|g_t(x) - \nabla f(x)\|^2\right] \leq \sigma^2\,,\end{cases}$$

for some $\sigma > 0$. Let L_A be the largest eigenvalue of A, and assume that $L_A \leq 1$ (our result generalizes to any L_A). Let $(x_t)_{t\geq 0}$ be generated with:

$$\forall t \geq 0\,, \quad x^{t+1} = x^t - \eta g_t(x^t)\,,$$

for a fixed stepsize $\frac{1}{2L} \geq \eta > 0$, and assume that all the iterates lie in $\mathcal{X}$. Assume that f is minimized over $\mathcal{X}$ at some interior point $x^\star$. We have for any $T > 0$:

$$\mathbb{E}\left[f(x^T) - f(x^\star)\right] \leq e^{-\eta\mu T}\left(f(x^0) - f(x^\star)\right) + \frac{\eta L\sigma^2}{\mu}.$$

For fixed $T > 0$, setting $\eta = \min\left(1/(2L), \frac{1}{\mu T}\ln(\frac{f_0\mu^2 T}{L\sigma^2})\right)$ gives:

$$\mathbb{E}\left[f(x^T) - f(x^\star)\right] \leq e^{-\frac{\mu}{2L}T}\left(f(x^0) - f(x^\star)\right) + \frac{L\sigma^2}{\mu^2 T}\ln\left(\frac{f_0\mu^2 T}{L\sigma^2}\right).$$

Thus, for fixed target precision $\varepsilon > 0$, using stepsize $\eta_\varepsilon = \min\left(\frac{\mu\varepsilon}{2L\sigma^2}, \frac{1}{2L}\right)$ and setting $T_\varepsilon = \lceil \ln\left(\varepsilon^{-1}(f(x^0) - f(x^\star))\right)\frac{1}{\eta_\varepsilon\mu}\rceil$, we have:

$$f\left(\frac{1}{T_\varepsilon}\sum_{t<T_\varepsilon} x^t\right) - f(x^\star) \leq \varepsilon\,,$$

$\left\|\frac{1}{S}\sum_{s=1}^S X_s\right\|_{\psi_2} \leq \frac{1}{S^2}\sum_{s=1}^S \|X_s\|_{\psi_2}$, and we apply this to the random variables $\hat{\mu}_i \top y$

with a number of oracle calls

$$T_\varepsilon \leq \max\left(\frac{2L\sigma^2}{\varepsilon\mu^2}, \frac{2L}{\mu}\right) \ln\left(\varepsilon^{-1}(f(x^0) - f(x^\star))\right).$$

Proof. For some $t \geq 0$, denoting $f_t = \mathbb{E}\left[f(x^{t+1}) - f(x^\star)\right]$, using relative smoothness, unbiasedness of the stochastic gradients and then relative strong convexity:

$$\begin{aligned} f_{t+1} - f_t &\leq -\eta\mathbb{E}\left[\left\|\nabla f(x^t)\right\|^2\right] + \frac{\eta^2 L}{2}\mathbb{E}\left[\|g_t\|_A^2\right] \\ &\leq -\eta\mathbb{E}\left[\left\|\nabla f(x^t)\right\|^2\right] + \frac{\eta^2 L L_A}{2}\mathbb{E}\left[\|g_t\|^2\right] \\ &\leq -\eta\left(1 - \frac{\eta L L_A}{2}\right)\mathbb{E}\left[\left\|\nabla f(x^t)\right\|^2\right] + \frac{\eta^2 L L_A \sigma^2}{2}. \end{aligned}$$

Using relative strong convexity of f, we have:

$$\begin{aligned} \left\|\nabla f(x^t)\right\|^2 &\geq \frac{1}{L_A}\left\|\nabla f(x^t)\right\|_A^2 \\ &\leq \frac{2\mu}{L_A} f_t, \end{aligned}$$

yielding, for $\eta < 1/(LL_A)$

$$f_{t+1} - f_t \leq -2\eta\frac{\mu}{L_A} f_t + \frac{\eta^2 L L_A \sigma^2}{2}.$$

Then, for some $T > 0$ and since $L_A \leq 1$, sum the above inequality multiplied by $(1 - \eta\mu)^{-t-1}$:

$$\begin{aligned} \sum_{0 \leq t \leq T-1} (1-\eta\mu)^{-t-1} f_{t+1} - (1-\eta\mu)^{-t} f_t &\leq \frac{\eta^2 L\sigma^2}{2} \sum_{0 \leq t \leq T-1} (1-\eta\mu)^{-t-1} \\ &\leq \frac{\eta^2 L\sigma^2}{2} \frac{(1-\eta\mu)^{-t-1}}{\eta\mu}, \end{aligned}$$

leading to the desired result. □

Chapter 9

Meta-learning of shared structures for personalized learning

In the previous chapters, we studied the limits of collaborative learning based on similarity between agents, and provided decentralized algorithms that used gradient filtering between agents. However, this leaves open the question of collaboratively learning without knowing these similarities in the general setting. In this chapter, we thus study for the same collaborative learning problem, a different set of assumptions and its role on an eventual collaborative speedup.

Motivated by multi-task and meta-learning approaches, we consider the problem of learning structure shared by tasks or users, such as shared low-rank representations or clustered structures. While all previous works focus on linear regression, we consider more general convex objectives, where the structural low-rank and cluster assumptions are expressed on the optima of each function. We show that under mild assumptions such as *Hessian concentration* and *noise concentration at the optimum*, rank and clustered regularized estimators recover such structure, provided the number of samples per task and the number of tasks are large enough. We then study the problem of recovering the subspace in which all the solutions lie, in the setting where there is only a single sample per task: we show that in that case, the rank-constrained estimator can recover the subspace, but that the number of tasks needs to scale exponentially large with the dimension of the subspace. Finally, we provide polynomial-time algorithms via nuclear norm constraints.

Notation for this chapter. As opposed to the previous chapter, we here do not consider the online setting, but directly study estimators computed from existing samples. There are thus no algorithms, and the tools used are more related to multi-task and meta learning communities. To highlight these differences, the notation used for models differ in this chapter: local models are referred to as $w_i \in \mathbb{R}^d$ instead of x_i.

9.1. Introduction

In this chapter, we study the problem of jointly learning n models for n agents, based on the local datasets of these users, while making the kind of assumption described above. More formally, each agent i indexed by $i \in [n]$ has a dataset of size m, $\{\xi_{ij}\}_{j\in[m]}$ with $\xi_{ij} \sim \mathcal{D}_i$ and $\mathcal{D}_i$ its local distribution, and wishes to minimize its *generalization error*, defined as for some loss $F_i : \mathbb{R}^d \times \Xi \to \mathbb{R}$:

$$f_i(w) = \mathbb{E}\left[F_i(w, \xi_i)\right], \quad w \in \mathbb{R}^d, \quad \xi_i \sim \mathcal{D}_i\,.$$

In order to capture the minimization of each local function and thus *personalization*, the objective we want to minimize then writes as:

$$f(W) = \frac{1}{n}\sum_{i=1}^{n} f_i(w_i)\,, \quad W = (w_1|\dots|w_n) \in \mathbb{R}^{d\times n}\,. \tag{9.1}$$

Based on the m samples per agent we have, we denote as $\mathcal{L}$ the empirical risk, that writes:

$$\mathcal{L}(W) = \frac{1}{nm} \sum_{(i,j)\in[n]\times[m]} F_i(w_i, \xi_{ij}), \quad W = (w_1|\dots|w_n) \in \mathbb{R}^{d\times n}. \tag{9.2}$$

By minimizing this empirical risk, we wish to obtain good minimizers for the true test loss (Equation (9.1)), under adequate assumptions. For our problem, the structural assumptions described above then read as follows. Note that Assumption 9.1.2 is in fact a subcase of Assumption 9.1.1.

Assumption 9.1.1 (Low rank representation)**.** *There exist* $\{w_i^\star\}_{i\in[n]}$ *such that for all* i $w_i^\star$ *minimizes* f_i*, and the rank of the matrix* $W^\star = (w_1^\star|\dots|w_n^\star) \in \mathbb{R}^{d\times n}$ *is at most* r*. Equivalently, there exist an orthonormal matrix* $U_\star \in \mathbb{R}^{d\times r}$ *and* $V^\star = (v_1^\star, \dots, v_n^\star) \in \mathbb{R}^{r\times n}$ *such that* $w_i^\star = U_\star v_i^\star$ *for all* $i \in [n]$*.*

Assumption 9.1.2 (Clustered clients)**.** *There exist* $\{w_i^\star\}_{i\in[n]}$*,* $\{c_s^\star\}_{s\in[r]}$ *and* $\tau^\star : [n] \to [r]$ *such that for all* i*,* $w_i^\star$ *minimizes* f_i *and* $w_i^\star = c^\star_{\tau^\star(i)}$*.*

The purpose of this chapter is then to study natural minimizers under these assumptions, for convex objectives f_i. Typical instances are *linear quadratic regression*, for which $\xi_{ij} = (x_{ij}, y_{ij})$ where $x_{ij} \sim \mathcal{N}(0, I_d)$ and $y_{ij} = \langle w_i^\star, x_{ij}\rangle + z_{ij}$ for independent *label noise* $z_{ij} \sim \mathcal{N}(0, \varepsilon^2)$, and $F_i(w, (x, y)) = \frac{1}{2}(w^\top x - y)^2$, or *generalized linear models* (GLMs) – that include classification with the logistic loss – for which $F_i(w, \xi_{ij} = (x_{ij}, y_{ij}) = \ell_i(\langle w, x_{ij}\rangle, y_{ij})$. Under the generality of our assumptions (that we develop in a later section), estimators that are specific to instances of our problem cannot be considered (such as Method of Moments (MoM) estimators [TJJ21, DFHT22] for linear quadratic regression). We hence restrict ourselves to **low-rank estimators** defined as, or that seek to approximate the following:

$$\hat{W} \in \arg\min \left\{ \mathcal{L}(W) \,\middle|\, \operatorname{rank}(W) \le r \right\},$$

and to counterparts of such estimators in the clustered setting. This chapter focuses on deriving generalization guarantees for such estimators, as well as for some approximations of these estimators.

9.1.1. Related works

Multi-task learning. Multi-task learning (MTL) is a general approach to improve generalization guarantees of single tasks knowledge of other tasks as an inductive bias [Car97]. On the theory side, some structural properties need to be made in order to show the advantage of MTL, such as a small variance between tasks [CBLPP22, DCGP19], a shared sparse support for multi-task feature recovery [AEP06, LPTG09], and finally a low rank structure that takes the form of a lower dimensional linear representation [BKF22].

Meta-Learning of linear representations. Motivated by empirical evidence in deep-learning tasks [BCV13, CHMS21], linear representations are a simple yet insightful proxy for theoretical works to model simple structures. We describe in this paragraph theoretical advances in estimating such linear representations: it is to be noted that all previous works focus on the *quadratic linear regression* setting, much simpler than ours, that fits a general convex framework. [TJJ21] shows it is impossible to estimate a representation of dimension $r \lll d$, if the total number of samples nm satisfies $nm = \mathcal{O}(rd)$. [RT11, DHK+21] study the lowrank estimator as we define it, and show that the representation is learnt if the total number of samples satisfies $nm \gg rd$ with $m \gg d$ samples per task: such a large number of samples per task is prohibitive, since under such assumptions collaboration is no longer needed.

Convex relaxations via nuclear norm regularization have then been studied [BKF22, RT11] and provably learn the representations provided that $n \gg d$ and $m \gg \sqrt{rd}$. Finally, a long line of work relaxes the rank constrain by factorizing the matrix W as $W = AB^\top$ where $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{n\times r}$, an approach known as *Burer-Monteiro factorization* [BM03, BM04]. [TJJ21] showed in the quadratic regression setting with random inputs, that the local minima induced by Burer-Monteiro factorization all led to estimating the linear representation, provided that $nm \gg r^4 d$ with $m \gg r^4$. Then, algorithms based on alterative minimizations [CHMS21, TJNO21] require $nm \gg r^2 d$ and $m \gg r^2$, together with an initialization with non-trivial alignment with the ground truth representation. Finally, algorithms specific to meta-learning such as MAML or ANIL [CMOS22, YBF23] or even FedAvg [CHMS22] can learn the representation if the model is parameterized using a Burer-Monteiro factorization if the number of samples is large enough, even if the rank is mispecified [YBF23]. All the aforementioned works study the linear quadratic regression setting with random inputs. Going beyond this simplified setting to learn linear representations is thus of upmost interest, and [CHS+23] is to our knowledge the first work to do so, by studying 2-layered ReLU networks in a multi-task setting. Our work also goes in this direction: we relax the assumptions to the general convex setting. Such relaxations have also been considered in the compressed sensing literature, such as [ANW12] that studied Iterative Thresholding algorithms under general restricted strong convexity and smoothness assumptions.

Learning clusters of clients. Clustering historically refers to partitioning data points into a few clusters, a task related to multi-class classification. It provides a theory-friendly framework to study and more importantly develop similarity based algorithms when in multi-task, collaborative, or federated learning, the agents are assumed to forms clusters of similar users [SMS20]. Clustered agents can be enforced by regularized objectives [MMRS20, JVB08, ZCY11], iterative clustering algorithms [GCYR20]. Providing generalization guarantees and a collaborative speedup under a cluster assumption for our problem is however a challenging problem. [GCYR20] proves that provided a sufficiently good initialization (with non-negligible signal, which amounts to assume that clusters are already learnt), the clusters can be learnt with a speedup proportional to clusters' population, while [EMS22a] build a similarity graph and obtains similar results without any assumption on the initialization, but with degraded dimension dependency.

9.1.2. Outline of this chapter

We first introduce our setting and assumptions in Section 9.2.1: the mild noise assumptions (concentration of the noise *at the optimum*) and the pointwise concentration of Hessians we introduce are a core contribution of this chapter. We study rank-constrained and clustered-contrained estimators in Section 9.3 to upper bound the sample complexity of minimizing each function and learning the global structure, with applications to *few-shot* learning on a new task in Section 9.3.4. We then study the recovery of the subspace in the degenerate case where $m = 1$ (only one sample per agent) via rank-constraint, and show that it is still feasible, but requires a larger number of tasks (that scales exponentially with r). Finally, we relax the rank constraint into a nuclear norm constraint, and provide statistical guarantees for the obtained estimator, that can be efficiently computed.

9.2. Setting, assumptions and notations

Notations. For $w \in \mathbb{R}^d$, $\|w\|^2 = \sum_{k=1}^d w_k^2$ is its squared Euclidean norm and $\|w\|_1 = \sum_{k=1}^d |w_k|$ is ℓ^1-norm. For $W \in \mathbb{R}^{d\times n}$ with singular values $\lambda_1, \dots, \lambda_p$, $\|W\|_F^2 = \sum_{(i,k)\in[n]\times[d]} W_{ki}^2 = \sum_{k=1}^p \lambda_k^2$ is its squared Frobenius norm, $\|W\|_{\mathrm{op}} = \max_{k\in[p]} |\lambda_k|$ is its operator norm, and $\|W\|_* = \sum_{k=1}^p |\lambda_k|$ is its nuclear norm. For some differentiable function $g : \mathcal{X} \to \mathbb{R}$

on $\mathcal{X} \subset \mathbb{R}^d$, we define its *Bregman divergence* $D_g : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ as: for all $w, w' \in \mathcal{X}$, $D_g(w, w') = g(w) - g(w') - \langle \nabla g(w'), w - w' \rangle$. I_d is the identity matrix in $\mathbb{R}^d$. For symmetric matrices A, B, $A \preceq B$ means that all the eigenvalues of $B - A$ are non-negative.

9.2.1. Main assumptions

We here describe the assumptions we make on local tasks f_i: note that we go well beyond the classical linear regression with quadratic loss setting, where local functions are assumed to be of the $f_i(w) = \sum_j (\langle w, x_{ij} \rangle - y_{ij})^2$, for noisy labels $y_{ij} = \langle x_{ij}, w_i^\star \rangle + \varepsilon_{ij}$ and random inputs and noise x_{ij}, ε_{ij}.

Assumptions on local functions f_i and F_i. Besides using Assumptions 9.1.1 and 9.1.2 alternatively, that implicitly assume that each f_i is lower-bounded and minimized at some $w_i^\star$, we will make use of the following regularity assumption on the local objectives f_i and losses F_i throughout the chapter. For all $i \in [n]$, $\|w_i^\star\| \leq B$, for some $B > 0$. Each f_i is assumed to be μ-**strongly convex** and L-**smooth** (with eventually $\mu \geq 0$, which amounts to convexity) [Bub15]: for all $w \in \mathbb{R}^d$, the inequality $\frac{\mu}{2}\|w - w'\|^2 \leq D_{f_i}(w, w') \leq \frac{L}{2}\|w - w'\|^2$, which can be equivalently written as $\mu I_d \preceq \nabla^2 f_i(w) \preceq L I_d$, if f_i is twice differentiable. We also assume that each F_i is convex in its first argument, and respectively that $\nabla^2 F_i$ and ∇F_i are $H-$ and $L'-$**Lipschitz** in their first arguments.

Noise at the optimum. We now need to assume that noise at the optimum is not too large, wihch we quantify through a noise constant $\sigma_\star^2$ as follows.

Assumption 9.2.1. *For all $i \in [n]$, the random variable $\nabla_w F_i(w_i^\star, \xi_i)$ for $\xi_i \sim \mathcal{D}_i$ is a $\sigma_\star^2$-multivariate subexponential random variable, in the sense that for all $w \in \mathbb{R}^d$, $\langle \nabla_w F_i(w_i^\star, \xi_i), w \rangle$ is $\sigma_\star^2 B\|w\|$ subexponential:*

$$\forall t \geq 0\,, \qquad \mathbb{P}\left(\langle \nabla_w F_i(w_i^\star, \xi_i), w \rangle \geq t\right) \leq \exp\left(-\frac{t}{\sigma_\star^2 B\|w\|}\right).$$

This assumption essentially describes the noise at the optimum. In the case of linear quadratic regressions *i.e.* if $\xi_{ij} = (x_{ij}, y_{ij})$ where $x_{ij} \sim \mathcal{N}(0, I_d)$ and $y_{ij} = \langle w_i^\star, x_{ij} \rangle + z_{ij}$ for independent *label noise* $z_{ij} \sim \mathcal{N}(0, \varepsilon^2)$, the gradient at the optimum reads $\nabla_w F_i(w_i^\star, \xi_{ij}) = z_{ij} x_{ij}$, and thus Assumption 9.2.1 holds for $\sigma_\star^2 = \varepsilon$. For GLMs, we have $\nabla_w F_i(w_i^\star, \xi_{ij}) = \ell_i'(\langle w_i^\star, x_{ij} \rangle, y_{ij}) x_{ij}$, so that the same holds.

Hessian concentration. Finally, we will assume that the Hessian of local functions concentrate fast enough as the number of samples increase, in the following way.

Assumption 9.2.2. *For all $i \in [n]$, for all $w, w_1, w_2 \in \mathbb{R}^d$, the random variable $\langle w_1, \nabla^2 F_i(w, \xi_i) w_2 \rangle$ is a $\sigma^2\|w_1\|\|w_2\|$-subexponential random variable:*

$$\forall t \geq 0\,, \qquad \mathbb{P}\left(\langle w_1, \nabla^2 F_i(w, \xi_i) w_2 \rangle \geq t\right) \leq \exp\left(-\frac{t}{\sigma^2\|w_1\|\|w_2\|}\right).$$

Such an assumption simply quantifies concentration of Hessians around their means, and will be used to control quantities such as $\frac{1}{nm}\sum_{ij}\langle w_i', \nabla^2 F_i(w_i, \xi_{ij}) w_i'' \rangle - \frac{1}{n}\sum_i \langle w_i', \nabla^2 f_i(w_i) w_i'' \rangle$. This assumption holds for linear quadratic regression and GLMs, for which respectively $\nabla^2 F_i(w_i, (x_{ij}, y_{ij}) = x_{ij} x_{ij}^\top$ and $\nabla^2 F_i(w_i, (x_{ij}, y_{ij}) = \ell_i''(\langle x_{ij}, w_i \rangle, y_{ij}) x_{ij} x_{ij}^\top$, provided that ℓ'' is bounded.

9.2.2. Principal angle distance

Under Assumption 9.1.1, provided that the low rank representation $U^\star$ is known, the problem becomes statistically easier. Therefore, one of the goals will be to learn this orthogonal matrix, and we thus need to introduce the right metric to quantify the distance between two orthonormal projections (or between two subspaces). For two subspaces of $\mathbb{R}^d$ of dimension r, the principal angles between these two subspaces quantify how close they are [YL16]. Let $\mathcal{V}, \mathcal{V}'$ be two linear subspaces of $\mathbb{R}^d$, of dimension r, and let P, P' be the respective orthogonal projections onto these subspaces. The matrix $(I_d - P)P'$ is of rank at most r, and its singular values satisfy $1 \geq \lambda_1 \geq \ldots \geq \lambda_r \geq 0$: there exists $\Theta = (\theta_1, \ldots, \theta_r) \in [0, \pi/2]^r$ such that $\lambda_i = \sin(\theta_i)$ for $i \in [r]$. We thus have $(I_d - P)P' = A \sin(\Theta) B^\top$, where $A, B \in \mathbb{R}^{d \times r}$ are orthonormal matrices. $\Theta = (\pi/2 \geq \theta_1 \geq \ldots \geq \theta_r \geq 0)$ are the *principal angles* between the two subspaces $\mathcal{V}, \mathcal{V}'$ (or equivalently, between their projections).

Definition 9.2.1 (Principal angle distance)**.** *Let $\mathcal{V}, \mathcal{V}'$ be two subspaces of $\mathbb{R}^d$ of dimension at most r, of principal angles Θ. We define the sine principal angle distances between $\mathcal{V}$ and $\mathcal{V}'$ as:*

$$\mathrm{dist}_F^2(\mathcal{V}, \mathcal{V}') = \sum_{k=1}^{r} \sin^2(\theta_k)\,, \quad \mathrm{dist}_\rho^2(\mathcal{V}, \mathcal{V}') = \sin^2(\theta_1)\,. \tag{9.3}$$

Equivalently, we will also write $\mathrm{dist}^2(P, P')$ for the same quantity. Note that we have $\mathrm{dist}_F^2 \leq r\mathrm{dist}_\rho^2$.

Now, for $U, U' \in \mathbb{R}^{d \times r}$ with orthonormal columns, $P = UU^\top$ and $P' = U'U'^\top$ are two orthogonal projectors onto subspaces of dimension r: we also write $\mathrm{dist}(U, U')$ for the corresponding distance between the subspaces.

9.2.3. Baselines without collaboration, in idealized cases, and structured regularization

The simple baseline we should compare ourselves to – and that we desire to outperform –, is the *no-collaboration* estimator $\hat{W}_{bad}$ defined as

$$\hat{W}_{bad} = (\hat{w}_1, \ldots, \hat{w}_n)\,, \quad \hat{w}_i \in \operatorname*{arg\,min}_{\|w\| \leq B} \frac{1}{m} \sum_{j=1}^{m} F_i(w, \xi_{ij})\,,$$

where the $\hat{w}_i$ are thus found without collaboration and yields a performance of

$$f(\hat{W}_{bad}) = \mathcal{O}\left(\sqrt{\frac{d}{m}}\right)\,, \qquad \text{or} \quad \frac{1}{n}\sum_{i=1}^{n} \|\hat{w}_i - w_i^\star\|_F^2 = \mathcal{O}\left(\frac{d}{m}\right)\,.$$

if f_i are respectively assumed to be convex or strongly convex. Such an estimator that does not use collaboration thus requires $m = \Omega(d)$ samples per agent [Wai19, *e.g.*, for linear regression]. Our ultimate goal is thus to leverage collaboration in order to obtain much faster statistical rates, and we therefore need to go beyond the basic estimator $\hat{W}_{bad} \in \arg\min \mathcal{L}(W)$.

The most natural way to do so is to enforce collaboration *via* some structural regularization, by making local models w_i interdependent. Under Assumption 9.1.1, we therefore want to find the following low-rank estimator:

$$\hat{W}_{\text{low rank}} \in \arg\min \left\{ \mathcal{L}(W) \,\middle|\, \mathrm{rank}(W) \leq r\,,\ \max_{i \in [n]} \|w_i\| \leq B \right\}, \tag{9.4}$$

while if we work under the clustered assumption Assumption 9.1.2, we are looking for:

$$\hat{W}_{\text{clustered}} \in \arg\min \left\{ \mathcal{L}(W_{C,\pi}) \,\middle|\, C \in \mathbb{R}^{d \times r}\,,\ \pi : [n] \to [r]\,,\ \max_{i \in [n]} \|w_i\| \leq B \right\}, \tag{9.5}$$

where for $C = (c_1|\dots|c_r) \in \mathbb{R}^{d\times r}$ and $\pi : [n] \to [r]$ we write $W_{C,\pi} \in \mathbb{R}^{d\times n}$ the matrix whose row is $c_{\pi(i)}$.

A first question arising is thus: how do these estimators behave *i.e.*, how many samples m and agents n are required to find the true minimizers $w_i^\star$? We answer these in Sections 9.3 and 9.4 However interesting it might be since it gives an optimistic baseline for our problem, answering this question may not be satisfactory: the estimators defined in Equations (9.4) and (9.5) are solutions of non-convex problems that may be NP hard to compute directly, although many heuristics exist to compute the estimators defined in Equations (9.4) and (9.5): hard-thresholding algorithms, Bureir-Monteiro factorisations, clustering algorithms, etc. We will thus resort to tricks that consist in convex relaxations of the problems (Section 9.5) in order to obtain polynomial-time algorithms.

9.3. Statistical bounds on rank-regularized and clustered estimators

In this section, we first provide statistical upper bounds on the performances of the low rank estimator defined in (9.4). [TJJ21, Theorem 5] states that for a particular instance of our problem (quadratic linear regression with isotropic gaussian data and noise), for a number of samples per user m and a number of user n, any estimator $\hat{U}$ of $U^\star$ that is based on these nm samples, must satisfy

$$\mathrm{dist}_\rho^2(\hat{U}, U^\star) = \Omega\left(\frac{rd}{nm}\right).$$

Hence, under Assumption 9.1.1 or Assumption 9.1.2 we need at least a total number of samples greater than rd in order to learn $U^\star$, and thus at least as much to learn $W^\star$. The bounds we prove next for the low rank estimator is optimal in the sense we recover $U^\star$ and $W^\star$ provided that $nm \gg rd$ (up to log factors), but it however requires m (the number of samples per task) to be larger than r to be non trivial: we will thus wonder how we can relax this assumption using the more restrictive clustered assumption and the associated estimator defined in Equation (9.5). Finally, we will prove in next section that if m is smaller than r (which is the case in the 1-sample per agent case), the low rank estimator may still recover the linear representation $U_\star$ but if and only if the number of agents is exponentially large, therefore making the problem much harder.

9.3.1. Preliminary lemmas

We first begin by stating the ingredients that lead to our generalization bounds on the lowrank and cluster estimators. Due to their generality, we believe they can be of independent interest. The estimators we study in this section take the form:

$$\hat{W} \in \operatorname*{arg\,min}_{W\in\mathcal{W}} \mathcal{L}(W)\,,$$

where we recall that $\mathcal{L}(W) = \frac{1}{nm}\sum_{(i,j)\in[n]\times[m]} F_i(w_i, \xi_{ij})\,, \quad W = (w_1|\dots|w_n) \in \mathbb{R}^{d\times n}$, as defined in Equation (9.2), and where $\mathcal{W}$ a subset of $\mathbb{R}^{d\times n}$ that contains $W^\star$ in its interior. What we can first notice then is that $\mathcal{L}(\hat{W}) \le \mathcal{L}(W^\star)$ by definition of $W^\star$, leading to:

$$D_{\mathcal{L}}(\hat{W}, W^\star) \le \langle \nabla\mathcal{L}(W^\star), W^\star - \hat{W}\rangle\,,$$

where D_g is the Bregman divergence of some function g. Recalling that $\mathbb{E}\left[\mathcal{L}(W)\right] = f(W) = \frac{1}{n}\sum_{i=1}^n f_i(w_i)$ for some fixed W and that $W^\star$ minimizes f, we thus obtain $f(\hat{W}) - f(W^\star) \le \langle \nabla\mathcal{L}(W^\star), W^\star - W\rangle + D_{f-\mathcal{L}}(W, W^\star)$, leading to:

$$f(\hat{W}) - f(W^\star) \le \sup_{W\in\mathcal{W}} |\langle \nabla\mathcal{L}(W^\star), W^\star - W\rangle| + \sup_{W\in\mathcal{W}} |D_{f-\mathcal{L}}(W, W^\star)|\,,$$

where we introduce the suprema in order to deal with the fact that $\hat{W}$ and $\mathcal{L}$ (the nm samples) are not independent. In order to prove that $\hat{W}$ performs well, we thus need to control both terms of the RHS. The first term is a noise term, since $\mathbb{E}\left[\nabla\mathcal{L}(W^\star)\right] = \nabla f(W^\star) = 0$, and we will use Assumption 9.2.1 to control the deviations (uniformly over $\mathcal{W}$). For the second term (the Bregman divergence of the difference), noting that $\mathbb{E}\left[D_{f-\mathcal{L}}(W, W^\star)\right] = D_0(W, W^\star) = 0$, we will bound the deviations using our Hessian concentration assumption (Assumption 9.2.2), uniformly over $\mathcal{W}$. The following proposition shows the type of concentration we require.

Lemma 9.3.1. *Assume that for some constants $\alpha, \beta > 0$ we have for all $W \in \mathcal{W}$:*

$$|\langle \nabla\mathcal{L}(W^\star), W^\star - W\rangle| \leq \alpha\sigma_\star^2 \sup_{i\in[n]} \|w_i - w_i^\star\|_2^2 + \sigma_\star^2\sqrt{\frac{\alpha}{n}\|W - W^\star\|_F^2 \sup_{i\in[n]} \|w_i - w_i^\star\|_2^2}\,,$$

and for all $(W, W') \in \mathcal{W}^2$,

$$\left|D_{f-\mathcal{L}}(W, W')\right| \leq \beta\sigma^2 \sup_{i\in[n]} \left\|w_i - w_i'\right\|_2^2 + \sigma^2\sqrt{\frac{\beta}{n}\|W - W'\|_F^2 \sup_{i\in[n]} \|w_i - w_i'\|_2^2}\,.$$

Then, if $4B^2 \geq \sup_{W,W'\in\mathcal{W}} \sup_{i\in[n]} \|w_i - w_i'\|_2^2$, we have:

$$f(\hat{W}) - f(W^\star) \leq 4B^2(\alpha\sigma_\star^2 + \beta\sigma^2 + \sqrt{\alpha}\sigma_\star^2 + \sqrt{\beta}\sigma^2)\,.$$

Furthermore, if each f_i is $\mu-$strongly convex,

$$\frac{1}{n}\sum_{i=1}^n \|\hat{w}_i - w_i^\star\|_2^2 \leq 16\mu^{-1}B^2(\alpha\sigma_\star^2 + \beta\sigma^2) + 64\mu^{-2}B^2(\alpha\sigma_\star^4 + \beta\sigma^4)\,.$$

Proof. We have:

$$\begin{aligned} f(\hat{W}) - f(W^\star) &\leq \sup_{W\in\mathcal{W}} |\langle \nabla\mathcal{L}(W^\star), W^\star - W\rangle| + \sup_{W\in\mathcal{W}} |D_{f-\mathcal{L}}(W, W^\star)| \\ &\leq 4B^2(\alpha\sigma_\star^2 + \beta\sigma^2) + 2B\sigma_\star^2\sqrt{\frac{\alpha}{n}\left\|W^\star - \hat{W}\right\|_F^2} + 2B\sigma^2\sqrt{\frac{\beta}{n}\left\|W^\star - \hat{W}\right\|_F^2}\,. \end{aligned}$$

For the first part, $\left\|W^\star - \hat{W}\right\|_F^2 \leq nB^2$. For the second part, we have $f(\hat{W}) - f(W^\star) \geq \frac{\mu}{2n}\left\|W^\star - \hat{W}\right\|_F^2$, leading to

$$\frac{1}{n}\left\|W^\star - \hat{W}\right\|_F^2 \leq 8\mu^{-1}B^2(\alpha\sigma_\star^2 + \beta\sigma^2) + 4\mu^{-1}B(\sqrt{\alpha}\sigma_\star^2 + \sqrt{\beta}\sigma^2)\sqrt{\frac{1}{n}\left\|W^\star - \hat{W}\right\|_F^2}\,.$$

Using $x^2 \leq ax + b$ and $x \geq 0$ implies $x^2 \leq 2a^2 + 2b$, we obtain that

$$\begin{aligned} \frac{1}{n}\left\|W^\star - \hat{W}\right\|_F^2 &\leq 16\mu^{-1}B^2(\alpha\sigma_\star^2 + \beta\sigma^2) + 32\mu^{-2}B^2(\sqrt{\alpha}\sigma_\star^2 + \sqrt{\beta}\sigma^2)^2 \\ &\leq 16\mu^{-1}B^2(\alpha\sigma_\star^2 + \beta\sigma^2) + 64\mu^{-2}B^2(\alpha\sigma_\star^4 + \beta\sigma^4) \end{aligned}$$

□

We are now armed, simply with this lemma combined to the two concentration lemmas below, to study estimators such as those defined in Equation (9.4) (low rank estimator) and Equation (9.5) (clustered estimator).

Lemma 9.3.2 (Noise concentration). *Let $\Delta \in \mathbb{R}^{d\times n}, t > 0$. We have:*

$$\mathbb{P}\left(|\langle \nabla \mathcal{L}(W^\star), \Delta\rangle| > t\right) \leq 2\exp\left(-c_1 \min\left(\frac{nmt}{\sigma_\star^2 \sup_{i\in[n]} \|\Delta_i\|^2}, \frac{nmt^2}{\frac{\sigma_\star^4}{n}\|\Delta\|_F^2 \sup_{i\in[n]}\|\Delta_i\|^2}\right)\right).$$

Proof. Remark that we have:

$$\langle \nabla \mathcal{L}(W^\star), \Delta\rangle = \frac{1}{nm}\sum_{i,j}\langle \nabla_w F_i(w_i^\star, \xi_{ij}), \Delta_i\rangle,$$

and using Assumption 9.2.1, $\langle \nabla_w F_i(w_i^\star, \xi_{ij}), \Delta_i\rangle$ are *i.i.d.* $\sigma_\star^2 \sup_i \|\Delta_i\|^2$ subexponential centered random variables. By Hanson-Wright inequality [HW71, RV13], we have our result. □

Lemma 9.3.3. *Let $W, W' \in \mathbb{R}^{d\times n}, t > 0$. We have:*

$$\mathbb{P}\left(\left|D_{f-\mathcal{L}}(W, W')\right| > t\right) \leq \frac{8H \sup_{i\in[n]}\|w_i - w_i'\|}{t} \times \exp\left(-c_2 \min\left(\frac{nmt}{\sigma^2 \sup_{i\in[n]}\|w_i - w_i'\|^2}, \frac{nmt^2}{\frac{\sigma^4}{n}\|W - W'\|_F^2 \sup_{i\in[n]}\|w_i - w_i'\|^2}\right)\right).$$

Proof. Using Taylor's second order equality, there exists $\tilde{W} = \lambda W + (1-\lambda)W' \in [W, W']$ (for $\lambda \in [0,1]$) such that

$$\begin{aligned} D_{\hat{\mathcal{L}}_{n,m}-f}(W, W') &= \frac{1}{2}\|W - W'\|^2_{\nabla^2(\hat{\mathcal{L}}_{n,m}-f)(\tilde{W})} \\ &= \frac{1}{nm}\sum_{ij}(w_i - w_i')^\top\left(\nabla^2_{ww}F_i(\tilde{w}_i, \xi_{ij}) - \nabla^2 f_i(\tilde{w}_i)\right)(w_i - w_i'). \end{aligned}$$

The subtlelty here lies in the fact that $\tilde{W}$ depends on the samples ξ_{ij}, preventing us from directly using concentration results. We will thus prove concentration of

$$\frac{1}{nm}\sum_{ij}(w_i - w_i')^\top\left(\nabla^2_{ww}F_i(\Delta_i, \xi_{ij}) - \nabla^2 f_i(\Delta_i)\right)(w_i - w_i'),$$

for some fixed $\Delta_i \in \mathbb{R}^{d\times n}$. $(w_i - w_i')^\top \nabla^2_{ww}F_i(\Delta_i, \xi_{ij})(w_i - w_i')$ is $\|w_i - w_i'\|^2\sigma^2$−subexponential (Assumption 9.2.2) and thus, by a centering Lemma, since

$$(w_i - w_i')^\top \nabla^2 f_i(\Delta_i)(w_i - w_i') = \mathbb{E}\left[(w_i - w_i')^\top \nabla^2_{ww}F_i(\Delta_i, \xi_{ij})(w_i - w_i')\right],$$

and thus $(w_i - w_i')^\top\left(\nabla^2_{ww}F_i(\Delta_i, \xi_{ij}) - \nabla^2 f_i(\Delta_i)\right)(w_i - w_i')$ is a $c\|w_i - w_i'\|^2\sigma^2$−subexponential random variable. Using Hanson-Wright inequality [HW71]:

$$\mathbb{P}\left(\left|\frac{1}{nm}\sum_{ij}(w_i - w_i')^\top\left(\nabla^2_{ww}F_i(\Delta_i, \xi_{ij}) - \nabla^2 f_i(\Delta_i)\right)(w_i - w_i')\right| > t\right) \leq 2\exp\left(-c_2 \min\left(\frac{nmt}{\sigma^4 \sup_{i\in[n]}\|w_i - w_i'\|^2}, \frac{nmt^2}{\frac{\sigma^4}{n}\|W - W'\|_F^2 \sup_{i\in[n]}\|w_i - w_i'\|^2}\right)\right).$$

We however want such a bound over $\sup_{\Delta\in[W,W']}\left|\frac{1}{nm}\sum_{ij}(w_i - w_i')^\top\left(\nabla^2_{ww}F_i(\Delta_i, \xi_{ij}) - \nabla^2 f_i(\Delta_i)\right)(w_i - w_i')\right|$. The quantity in the sup here is $2H\sup_{i\in[n]}\|w_i - w_i'\|$ Lipschitz in Δ_i. Since $\{W + \lambda(W' - W), \lambda \in [0,1]\} =$

$[W, W']$,taking an $\varepsilon-$net of $[0,1]$ (of size $\leq 2/\varepsilon$),

$$\mathbb{P}\Bigg(\forall \Delta \in [W, W'], \quad \left|\frac{1}{nm}\sum_{ij}(w_i - w_i')^\top \left(\nabla^2_{ww} F_i(\Delta_i, \xi_{ij}) - \nabla^2 f_i(\Delta_i)\right)(w_i - w_i')\right| \\ > t + 2H \sup_{i\in[n]} \left\|w_i - w_i'\right\|\varepsilon\Bigg) \\ \leq \frac{4}{\varepsilon}\exp\left(-c_2 \min\left(\frac{nmt}{\sigma^4 \sup_{i\in[n]}\|w_i - w_i'\|^2}, \frac{nmt^2}{\frac{\sigma^4}{n}\|W - W'\|_F^2 \sup_{i\in[n]}\|w_i - w_i'\|^2}\right)\right).$$

Thus, for $\varepsilon = \frac{t}{2H \sup_{i\in[n]}\|w_i - w_i'\|}$, we have our result.

□

9.3.2. Generalization properties of the low-rank estimator

Although the underlying non-convex optimization problem does not seem possible to solve in polynomial time, efficient approximations exist. One approach is the Burer-Monteiro factorization [BM03, BM04], which involves parameterizing A_k as $A_k = B_k C_k$ with $B_k \in \mathbb{R}^{d\times r}$ and $C_k \in \mathbb{R}^{r\times d}$. This method relaxes the constraint to a convex set but results in a non-convex function. Another approach is *hard-thresholding*, which use projected (S)GD on the non-convex constraint set [BD09, FS18].

Theorem 9.1. *Assume that Assumption 9.1.1 holds (underlying low rank assumption). Then, with probability* $1 - 4e^{-r(d+n)} - 4e^{-nmd}$,

$$f(\hat{W}_{\text{low rank}}) - f(W^\star) = \tilde{\mathcal{O}}\left(B^2(\sigma^2 + \sigma_\star^2)\sqrt{\frac{r(d+n)}{mn}}\right).$$

and, if each f_i *is* $\mu-$*strongly convex with* $\mu > 0$,

$$\frac{1}{n}\left\|\hat{W}_{\text{low rank}} - W^\star\right\|_F^2 = \tilde{\mathcal{O}}\left(B^2(\sigma^4 + \sigma_\star^4)\frac{r(d+n)}{\mu^2 mn}\right).$$

The low rank estimator hence only needs $nm = \tilde{\Omega}(rd)$ samples, under the condition $m = \tilde{\Omega}(r)$, in order to recover good performances. In the strongly convex case, we recover all the $w_i^\star$, and thus we can recover the subspace, reaching the optimal rd sample complexity, but with the additional $m \gg r$ condition, which is expected since without this condition, recovering the local heads is impossible. The dependency on d is expected, but can easily be replaced by an *effective dimension* d_{eff} using more refined concentration results for Hessians [EM21]. One question that arises is then: can we hope to recover the underlying subspace even for small $m < r$? We do this in next subsection, by further assuming that agents form clusters of agents (this is a particular instance of low rank). Then, in Section 9.4, we show that in a special case (noiseless linear regression with quadratic loss), we can recover the subspace even with $m = 1$.

9.3.3. Generalization properties of the clustered estimator

Recall that we defined the clustered estimator $\hat{W}_{\text{clustered}}$ in Equation (9.5), the underlying optimization problem being still non-convex. The following Theorem is proved exactly as in the low rank case.

Theorem 9.2. *Assume that Assumption 9.1.2 holds (underlying low rank assumption). Then, with probability* $1-4e^{-n}-4e^{-nmd}$, *we have* $f(\hat{W}_{\text{clustered}}) - f(W^\star) = \tilde{\mathcal{O}}\left(B^2(\sigma^2 + \sigma_\star^2)\sqrt{\frac{rd}{mn} + \frac{1}{m}}\right)$,

and if each f_i is $\mu-$strongly convex, we have $\frac{1}{n}\left\|\hat{W}_{\text{clustered}} - W^\star\right\|_F^2 = \tilde{\mathcal{O}}\left(B^2(\sigma^4+\sigma_\star^4)\frac{1}{\mu^2}\left(\frac{rd}{mn}+\frac{1}{m}\right)\right)$.

In this case, we thus only need $mn \gg rd$ and $m \gg 1$ in order to obtain good generalization, therefore bypassing the $m \gg r$ constraint of the lowrank estimator. However, due to our proof techniques, this was expected. Indeed, in Lemma 9.3.1, the uniform upper bounds on the quantities considered scale with $\mathcal{W}$ the set over which we optimize over: the large this set, the worse the bounds. Hence, restricting the set from low rank matrices to clustered matrices, we expect better uniform upper bounds. Quantitatively, these uniform upper bounds depend on the metric entropy of the sets considered [Lor66, Dud74]. For low rank matrices as considered in Theorem 9.1, the minimum size of an $\varepsilon-$net of the search set scales as $\left(\frac{9}{\epsilon}\right)^{cr(d+n)}$ [CP11], leading to an $\varepsilon-$entropy of $\mathcal{O}(r(d+n)\ln(1/\varepsilon))$. Dividing this metric entropy by the number of samples, we obtain $\frac{rd}{nm}+\frac{r}{m}$, a bound that cannot vanish for $m<r$. However, in the case of clustered matrices, we can see the search set as $[r]^{[n]}\times\mathbb{R}^{d\times r}$, leading to an $\varepsilon-$ entropy of $n\ln(r)+\mathcal{O}(rd\ln(1/\varepsilon))$. Dividing by the number of samples, we hence obtain a bound of order $\frac{rd}{nm}+\frac{\ln(r)}{m}$: for r such that $\ln(r)\lll r$ (*i.e.* for $r\gg 1$), we can thus obtain vanishing bounds even for $m \lll r$.

9.3.4. Few-shot learning on a new task and meta-learning of the linear representation

Under both Assumption 9.1.1 and 9.1.2, there respectively exist matrices $U^\star, C^\star \in \mathbb{R}^{d\times r}$ such that users optima can be written as $w_i^\star = U^\star v_i^\star$ for *local heads* $v_i^\star\in\mathbb{R}^r$ and $w_i^\star = C^\star P_i^\star$ for $P^\star\in\mathbb{R}^{r\times n}$ local cluster identification. In both cases, learning this $d\times r$ matrix simplifies the problem: learning them makes the problem much easier for learning a new task that shares the same structure. The following result [TJJ21, Lemma 16] shows that when we learn $W^\star$, we indeed learn this shared representation, as expected.

Lemma 9.3.4. *Assume that Assumption 9.1.1 holds and that we have access to some $\hat{W}\in\mathbb{R}^{d\times n}$ of rank at most r such that $\frac{1}{n}\left\|\hat{W}-W^\star\right\|_F^2\le\varepsilon$. Then, writing $\hat{W}=\hat{U}\hat{V}$ for $\hat{U}\in\mathbb{R}^{d\times r}$ with orthogonal columns, we have $\frac{\text{dist}_F^2(\hat{U},U^\star)}{r}\le\frac{\varepsilon}{\nu^2}$, where $\nu^2=\frac{r}{n}\sigma_r(V^\star V^{\star,\top})$ is the smallest eigenvalue of $\frac{r}{n}V^\star V^{\star,\top}=\frac{r}{n}\sum_{i=1}^n v_i^\star v_i^{\star,\top}=\in\mathbb{R}^{r\times r}$.*

Note that for well-conditioned matrix $W^\star$, we expect $\nu^2=\Omega(1)$ (this is the case for instance if we choose $v_i^\star\sim\mathcal{N}(0,I_r/r)$). Hence, if the active n agents learn their local optima, the whole pool of agents has access to the shared representation $U^\star$, up to a small error δ. Then, a new agent may take advantage of this to perform *few-shot learning*: with only a few samples, the $(n+1)-$th agent may learn its local optimizer, as we show next.

Proposition 9.3.1. *Assume that Assumption 9.1.1 holds and that based on the nm samples $(\xi_{ij})_{i\in[n],j\in[m]}$ of the n users, an estimator $\hat{U}$ of $U^\star$ satisfying $\text{dist}_\rho^2(\hat{U},U^\star)\le\delta^2$ has been learnt. Let a $(n+1)-$th agent have m' samples $(\xi_{n+1,j})_{j\in[m']}$ from some distribution $\mathcal{D}_{n+1}$ that desires to minimize its local generalization error $f_{n+1}(w)=\mathbb{E}\left[F_{n+1}(w,\xi_{n+1})\right],\xi_{n+1}\sim\mathcal{D}_{n+1}$. Assume that f_{n+1}, F_{n+1} satisfy the assumptions of Section 9.2.1 and that f_{n+1} is minimized at some $w_{n+1}^\star=U^\star v_{n+1}^\star$ for some $v_{n+1}^\star\in\mathbb{R}^r$ that satisfies $\left\|v_{n+1}^\star\right\|\le B$. Then, with probability $1-e^{-m'r}$, the estimator*

$$\hat{v}_{n+1}\in\arg\min\left\{\frac{1}{m'}\sum_{j=1}^{m'}F_{n+1}(\hat{U}v,\xi_{n+1,j})\,|\,v\in\mathbb{R}^r\,,\,\|v\|\le B\right\},\tag{9.6}$$

satisfies $f_{n+1}(\hat{U}\hat{v}_{n+1})-f_{n+1}(W_{n+1}^\star)=\frac{L\delta^2}{2}+\tilde{\mathcal{O}}\left(B^2(\sigma^2+\sigma_\star^2)\sqrt{\frac{r}{m'}}\right)$, and if f_{n+1} is $\mu-$strongly convex, $\left\|\hat{v}_{n+1}-v_{n+1}^\star\right\|^2=\frac{L\delta^2}{\mu}+\tilde{\mathcal{O}}\left(\frac{B^2}{\mu^2}(\sigma^4+\sigma_\star^4)\frac{r}{m'}\right)$.

Hence, if a reprensentation has been learnt with enough precision, an incoming agent only needs $m' \gg r$ samples to perform *few-shot learning*. However, as highlighted by our previous bounds and in particular by Equation (9.4), up to now the representation is only learnt by learning beforehand the whole matrix $W^\star$ of concatenated heads. The following question thus arises: *can we meta-learn the linear representation $U^\star$ without learning local heads $w_i^\star$?* Aside from the fact that having many tasks with few samples is often easier than having more samples from existing tasks, this question has other interests in practice. For instance, being able to learn the representation with only one sample per agent will inherently be more private, since then users only need to share a single random sample from their dataset that is thus kept private.

9.4. Low-rank estimators with only 1 sample per user

We here study under Assumption 9.1.1 if the linear representation $U^\star$ can be learnt if we only have 1 sample per agent: $m = 1$. This case lies in the *meta-learning-without-learning* setting: for each task, it is illusory to want to minimize the local objective f_i and to find $w_i^\star$ wether or not a low dimensional linear representation is learnt; yet learning this representation may help for future tasks, and quite often it is easier to obtain many tasks with few samples per task, than tasks with many samples. However, this setting is quite challenging, since no concentration result will hold. For presentation sake we focuse on $m = 1$ in this section; however, our arguments easily generalize to $m \leq (1-\varepsilon)r$ for $\varepsilon > 1/r$. To simplify the analysis, we here fall back to studying noisy linear regression with quadratic loss, for which $F_i(w, \xi_{ij} = (x_{ij}, y_{ij})) = \frac{1}{2}(\langle x_i, w\rangle - y_i)^2$, where $\xi_{ij} = (x_{ij}, y_{ij})$ for $y_{ij} = \langle w_i^\star, x_{ij}\rangle$ and $x_{ij} \sim \mathcal{N}(0, I_d)$. We furthermore assume that there exists $\lambda > 0$ such that $\frac{1}{n}\sum_{i=1}^n w_i^\star w_i^{\star\top} \geq \frac{\lambda B^2}{r} U^\star U^{\star\top}$, and that all $w_i^\star$ are of norm B. For $\lambda = \Omega(1)$ this means that the local heads span all directions of the low-rank subspace.

Theorem 9.3. *Assume that Assumption 9.1.1 holds. Let $\hat{W}$ be any solution of the optimization problem defined in Equation (9.4), and let $\hat{U} \in \mathbb{R}^{d\times r}$ be the orthogonal projection on its image. Let $\varepsilon \in (0,1)$. There exist constants $c, C > 0$ such that with probability $1 - e^{-rdn}$,*

$$\text{if} \quad n \geq Cd\ln(d)e^{\frac{cr}{\lambda\varepsilon}}\,, \quad \text{then} \quad \frac{1}{r}\mathrm{dist}_F^2(\hat{U}, U) \leq \varepsilon\,.$$

Proof of Theorem 9.3. In this proof, we assume that there is only one sample per task: $m = 1$. To simplify notations, we thus write $y_i = \langle w_i^\star, x_i\rangle$. The following notion of "admissibility" for some orthogonal matrix is then introduced: a matrix is said admissible if it satisfies the same properties as $U^\star$.

Definition 9.4.1. *We say that some orthogonal matrix $U \in \mathbb{R}^{d\times r}$ is δ-admissible for $\delta \in (0,1]$ if there exists $v_1, \ldots, v_n \in \mathbb{R}^r$ such that for all $i \in [n]$ we have $y_i = \langle Uv_i, x_i\rangle$ and $\delta\|v_i\| \leq B$.*

We now compute the probability that some given matrix U is admissible.

Proposition 9.4.1. *Assume that the Gaussian random design holds. Let $U \in \mathbb{R}^{d\times r}$ be an orthogonal matrix, let $\varepsilon = \mathrm{dist}_F^2(U, U^\star)$, and assume that $\varepsilon > 0$. Furthermore, assume that (i) for all $i \in [n]$, $\|w_i^\star\| = B$ and (ii) there exists $\lambda > 0$ such that $\frac{1}{n}\sum_{i=1}^n w_i^\star w_i^{\star\top} \geq \frac{\lambda B^2}{r} U^\star U^{\star\top}$. Then there exist constants $c_1, c_2 > 0$ such that if $\delta \geq 1 - \frac{\lambda\varepsilon}{4}$, we have:*

$$\mathbb{P}\left(U \text{ is } \delta\text{-admissible}\right) \leq \exp\left(-c_1 n e^{-\frac{c_2 r}{\lambda\varepsilon}}\right)\,.$$

Proof. Notice the following.

Lemma 9.4.1. *We have the following equivalence:*

$$U \text{ is } \delta\text{-admissible} \quad \iff \quad \forall i \in [n]\,, \quad \left\|U^\top x_i\right\|_2 \geq \frac{\delta|y_i|}{B}\,.$$

Thus, $\mathbb{P}\left(U \text{ is } \delta\text{-admissible}\right) = \prod_{i=1}^n \mathbb{P}\left(\left\|U^\top x_i\right\|_2 \geq \frac{\delta|y_i|}{B}\right)$, by independence of the samples. We have $y_i = \langle Uv_i^\star, x_i\rangle$, leading us to consider the decomposition $U^\top = U^\top P_i^\star + U^\top(I_d - P_i^\star)$, where $P_i^\star = p_i^\star p_i^{\star\top} \in \mathbb{R}^{d\times d}$ is the orthogonal projector onto the span of $w_i^\star$, for $p_i^\star = \frac{w_i^\star}{\|w_i^\star\|_2}$. Thus, $\left\|U^\top x_i\right\|_2 = \left\|U^\top P_i^\star x_i + U^\top(I_d - P_i^\star)x_i\right\|_2 \leq \left\|U^\top P_i^\star x_i\right\|_2 + \left\|U^\top(I_d - P_i^\star)x_i\right\|_2$. We now assume for simplicity that $B = 1$. We have:

$$\begin{aligned}
\mathbb{P}\left(\left\|U^\top x_i\right\|_2 \geq \frac{\delta|y_i|}{B}\right) &\leq \mathbb{P}\left(\left\|U^\top(I_d - P_i^\star)x_i\right\|_2 \geq \delta|\langle w_i^\star, x_i\rangle| - \left\|U^\top P_i^\star x_i\right\|_2\right) \\
&\leq \mathbb{P}\left(\left\|U^\top(I_d - P_i^\star)x_i\right\|_2 \geq \delta|\langle w_i^\star, x_i\rangle| - \left\|U^\top p_i^\star\right\|_2|\langle p_i^\star, x_i\rangle|\right) \\
&= \mathbb{P}\left(\left\|U^\top(I_d - P_i^\star)x_i\right\|_2 \geq \left(\delta - \frac{\left\|U^\top p_i^\star\right\|_2}{\|w_i^\star\|_2}\right)|\langle w_i^\star, x_i\rangle|\right)\,,
\end{aligned}$$

since $U^\top P_i^\star x_i = \langle p_i^\star, x_i\rangle U^\top p_i^\star = \frac{1}{\|w_i^\star\|}\langle w_i^\star, x_i\rangle U^\top p_i^\star$.

We now notice that $\left\|U^\top(I_d - P_i^\star)x_i\right\|_2$ and $|\langle w_i^\star, x_i\rangle|$ are independent random variables, using the Gaussian random design assumption and the fact that $w_i^\star$ is by definition in the orthogonal of the span of $(I_d - P_i^\star)$. We have $U^\top(I_d - P_i^\star)x_i \sim \mathcal{N}(0, U^\top(I_d - P_i^\star)U)$ and since $U^\top(I_d - P_i^\star)U \preceq U^\top U = I_r$, the random variable $\left\|U^\top(I_d - P_i^\star)x_i\right\|_2^2$ is stochastically dominated by Z_i a χ_r^2 random variable of dimension r independent of $\langle w_i^\star, x_i\rangle$. Then, writing $z_i = \langle w_i^\star, x_i\rangle^2$ (a χ_1^2 random variable), we have that z_i and Z_i are independent, and:

$$\mathbb{P}\left(\left\|U^\top x_i\right\|_2 \geq \delta|y_i|\right) \leq \mathbb{P}\left(Z_i \geq \alpha_i z_i\right)\,,$$

where $\alpha_i = \max\left(0, \delta - \frac{\left\|U^\top p_i^\star\right\|_2}{\|w_i^\star\|_2}\right)^2$. To upper-bound $\mathbb{P}\left(Z_i \geq \alpha_i z_i\right)$, we lower bound $\mathbb{P}\left(Z_i \leq \alpha_i z_i\right)$. For $Z_i = a_1 + \ldots + a_r$ where (a_i) are independent χ_1^2 random variables,

$$\begin{aligned}
\mathbb{P}\left(Z_i \leq \alpha_i z_i\right) &\geq \mathbb{P}\left(Z_i \leq r/2, \alpha_i z_i \geq r/2\right) \\
&= \mathbb{P}\left(Z_i \leq r/2\right)\mathbb{P}\left(\alpha_i z_i \geq r/2\right) \\
&\geq \mathbb{P}\left(a_1 \leq 1/2\right)^r \mathbb{P}\left(\alpha_i z_i \geq r/2\right) \\
&\geq \mathbb{P}\left(a_1 \leq 1/2\right)^r \frac{2}{r\alpha_i} e^{-\frac{r}{4\alpha_i}} \\
&\geq \frac{2}{r\alpha_i} e^{-\frac{c'r}{\alpha_i}} \\
&\geq e^{-\frac{cr}{\alpha_i}}\,.
\end{aligned}$$

Thus,

$$\mathbb{P}\left(\left\|U^\top x_i\right\|_2 \geq \frac{\delta|y_i|}{B}\right) \leq 1 - e^{-\frac{cr}{\alpha_i}}\,,$$

and

$$\mathbb{P}\left(U \text{ is } \delta\text{-admissible}\right) \leq \prod_{i=1}^n \left(1 - e^{-\frac{cr}{\alpha_i}}\right)\,,$$

for $\alpha_i = \max\left(0, \delta - \frac{\|U^\top p_i^\star\|_2}{\|w_i^\star\|_2}\right)^2$.

In order to use this without too much trouble, we now make the following assumptions about the norm of each vector $w_i^\star$ and about the diversity of the direction they span: *(i)* for all $i \in [n]$, $\|w_i^\star\|_2 = 1$ and *(ii)* $\frac{1}{n}\sum_{i=1}^n p_i^\star p_i^{\star\top} \preceq \frac{L}{r} U^\star U^{\star\top}$ for some $\lambda > 0$.

Our goal is now to lower bound the α_i's. Using *(i)*, we can simplify each α_i as $\alpha_i = \max\left(0, \delta - \left\|U^\top p_i^\star\right\|_2\right)^2$. Then, we have

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^n \left\|U^\top p_i^\star\right\|_2^2 &= \frac{1}{n}\sum_{i=1}^n p_i^{\star\top} UU^\top p_i^\star \\
&= \frac{1}{n}\sum_{i=1}^n p_i^{\star\top} p_i^\star - \frac{1}{n}\sum_{i=1}^n p_i^{\star\top}\left(U^\star U^{\star\top} - UU^\top\right) p_i^\star \\
&= 1 - \frac{1}{n}\sum_{i=1}^n \mathrm{Tr}\Big(p_i^{\star\top}\left(I_d - UU^\top\right) p_i^\star\Big) \\
&= 1 - \frac{1}{n}\sum_{i=1}^n \mathrm{Tr}\Big(p_i^\star p_i^{\star\top}\left(I_d - UU^\top\right)\Big) \\
&= 1 - \mathrm{Tr}\Big(\frac{1}{n}\sum_{i=1}^n p_i^\star p_i^{\star\top}\left(I_d - UU^\top\right)\Big) \\
&\leq 1 - \frac{\lambda}{r}\mathrm{Tr}\Big(U^\star U^{\star\top}\left(I_d - UU^\top\right)\Big) \\
&= 1 - \frac{\lambda}{r}\mathrm{Tr}\Big(\left(I_d - UU^\top\right) U^\star U^{\star\top}\Big) \\
&= 1 - \frac{\lambda}{r}\mathrm{dist}_F^2(U, U^\star) \\
&= 1 - \lambda\varepsilon\,,
\end{aligned}$$

where we used $\mathrm{dist}_F^2(U, U^\star) = \|(I-P)P^\star\|_F^2 = \mathrm{Tr}((I-P)P^\star((I-P)P^\star)^\top) = \mathrm{Tr}((I-P)P^\star(I-P)) = \mathrm{Tr}((I-P)P^\star)$, where $P = UU^\top$ and $P^\star = U^\star U^{\star\top}$. Hence, since for all i we have $\left\|U^\top p_i^\star\right\|_2^2 \leq 1$, using $\frac{1}{n}\sum_{i=1}^n \left\|U^\top p_i^\star\right\|_2^2 \leq 1 - \lambda\varepsilon$, we have that:

$$\begin{aligned}
1 - \lambda\varepsilon &\geq \frac{1}{n}\sum_{i=1}^n \left\|U^\top p_i^\star\right\|_2^2 \\
&\geq (1 - \lambda\varepsilon/2)\frac{|\Big\{i : \left\|U^\top p_i^\star\right\|_2^2 \geq 1 - \lambda\varepsilon/2\Big\}|}{n} \\
&= (1 - \lambda\varepsilon/2)\left(1 - \frac{|\Big\{i : \left\|U^\top p_i^\star\right\|_2^2 \leq 1 - \lambda\varepsilon/2\Big\}|}{n}\right),
\end{aligned}$$

so that $|\Big\{i : \left\|U^\top p_i^\star\right\|_2^2 \leq 1 - \lambda\varepsilon/2\Big\}| \geq \left(1 - \frac{1-\lambda\varepsilon}{1-\lambda\varepsilon/2}\right)n = \frac{\lambda\varepsilon/2}{1-\lambda\varepsilon/2}n \geq \frac{\lambda\varepsilon n}{2}$: at least $\frac{\lambda\varepsilon n}{2}$ of the α_i's are thus smaller than $\max(0, \frac{\lambda\varepsilon}{2} - (1-\delta)) \leq \frac{\lambda\varepsilon}{4}$ if $\delta \geq 1 - \frac{\lambda\varepsilon}{4}$. This leads to:

$$\mathbb{P}\left(U \text{ is } \delta\text{-admissible}\right) \leq \prod_{i=1}^n \left(1 - e^{-\frac{cr}{\alpha_i}}\right) \leq \left(1 - e^{-\frac{C'r}{\lambda\varepsilon}}\right)^{\frac{\lambda\varepsilon n}{2}} \leq \exp\left(-c_1 n e^{-\frac{c_2 r}{\lambda\varepsilon}}\right).$$

□

Now that we have proved Proposition 9.4.1, we can state our feasibility bound on the

number of tasks when there is only one sample per task. The estimator we build is quite simple. We define it as any $\hat{U}$ that satisfies

$$\hat{U} \in \left\{ U \in \mathbb{R}^{d\times r} \text{ orthonormal such that } \forall i \in [n]\,, \quad B\left\|U^\top x_i\right\| \geq |y_i| \right\},$$

and this set is not empty since it contains $U^\star$. In fact, any $\hat{U}$ built from the image of some

$$\hat{W} \in \arg\min \left\{ \mathcal{L}(W)\,, \quad \operatorname{rank}(W) \leq r\,, \quad \|w_i\|_2 \leq B \right\},$$

is in this set.

We finally conclude by proving our theorem. Let $\mathcal{U} = \left\{ U \in \mathbb{R}^{d\times r} : U \text{ is orthonormal}, \operatorname{dist}_F^2(U, U^\star) \geq \varepsilon \right\}$ and let $\mathcal{U}_\eta$ be an η-net of $\mathcal{U}$ of minimal cardinality. We know that we have $|\mathcal{U}_\eta| \leq e^{crd\ln(1/\eta)}$. For $U \in \mathcal{U}$, there exists $U_\eta \in \mathcal{U}_\eta$ such that $\|U - U_\eta\| \leq \eta$, and for any $i \in [n]$,

$$\left\|U^\top x_i\right\| \leq \left\|U_\eta^\top x_i\right\| + \left\|(U - U_\eta)^\top x_i\right\| \leq \left\|U_\eta^\top x_i\right\| + \eta\|x_i\|\,.$$

Thus, if U_η is not δ-admissible for some δ, there exists i such that $\left\|U_\eta^\top x_i\right\| \leq \delta|y_i|$, and thus

$$\left\|U^\top x_i\right\| \leq \delta|y_i| + \eta\|x_i\| < |y_i|\,,$$

provided that $\eta < \frac{|y_i|}{\|x_i\|}(1-\delta)$.

Let thus $\delta = 1 - \frac{\lambda\varepsilon}{4}$ and $\eta < \min_{i\in[n]} \frac{|y_i|}{\|x_i\|}(1-\delta)$. With high probability, we have η of order $\frac{1}{\sqrt{d}}\lambda\varepsilon$ (up to constant and log factors). As above, let $\mathcal{U}_\eta$ be an η-net of $\mathcal{U}$ (of cardinality less than $e^{crd\ln(1/\eta)}$). Using an union bound over $\mathcal{U}_\delta$ and proposition 9.4.1:

$$\begin{aligned}\mathbb{P}\left(\exists U \in \mathcal{U}_\delta \text{ admissible}\right) &\leq |\mathcal{U}_\delta| \exp\left(-c_1 n e^{-\frac{c_2 r}{\lambda\varepsilon}}\right) \\ &\leq \exp\left(crd\ln(1/\delta) - c_1 n e^{-\frac{c_2 r}{\lambda\varepsilon}}\right)\,.\end{aligned}$$

Hence, if $n \geq \frac{e^{\frac{c_2 r}{\lambda\varepsilon}}}{c_1} rd(1 + c\ln(1/\delta))$, we have that with probability $1 - e^{-rdn}$, all elements of $\mathcal{U}_\delta$ are not admissible.

We now place ourselves on this event of probability $1 - e^{-rnd}$. Let $U \in \mathcal{U}$. In light of section 9.4, there exists $i \in [n]$ such that $\left\|U^\top x_i\right\| \leq y_i$: U is not admissible. Since $U^\star$ is 1-admissible, the set $\left\{ U \in \mathbb{R}^{d\times r} \text{ orthonormal and admissible} \right\}$ is non empty. Let thus $\hat{U}$ be any matrix satisfying

$$\hat{U} \in \left\{ U \in \mathbb{R}^{d\times r} \text{ orthonormal such that } \forall i \in [n]\,, \quad \left\|U^\top x_i\right\| \geq |y_i| \right\}.$$

Since this set does not intersect $\mathcal{U}$, we have $\frac{1}{r}\operatorname{dist}_F^2(\hat{U}, U^\star) \leq \varepsilon$, concluding our proof.

We now prove Proposition 9.4.2. Without loss of generality, assume that $B = 1$. Let $U \in \mathbb{R}^{d\times r}$ with orthogonal columns such that the span of U is in the orthogonal of the span of $U^\star$. We have that:

$$\mathbb{P}\left(U \text{ is admissible}\right) = \mathbb{P}\left(\forall i \in [n], \left\|U^\top x_i\right\|_2^2 \geq y_i^2\right)\,.$$

Since span of U is in the orthogonal of the span of $U^\star$, $\left\|U^\top x_i\right\|_2^2$ and y_i^2 are independent random variables, of respective laws χ_r^2 and χ_1^2. This leads to:

$$\mathbb{P}\left(U \text{ is admissible}\right) = \mathbb{P}\left(\chi_r^2 \geq \chi_1^2\right)^n$$

$$\geq \mathbb{P}\left(\chi^2_r \geq r/2\right)^n \mathbb{P}\left(\chi^2_1 \leq r/2\right) .$$

Then, by χ^2-concentration, $\mathbb{P}\left(\chi^2_r \geq r/2\right) = 1 - \mathbb{P}\left(\chi^2_r - r < -r/2\right) \geq 1 - e^{-c_1 r}$. Similarly, $\mathbb{P}\left(\chi^2_1 \leq r/2\right) \geq 1 - e^{-c_2 r}$, leading to:

$$\begin{aligned}\mathbb{P}\left(U \text{ is admissible}\right) &\geq (1 - e^{-c_1 r})^n (1 - e^{-c_2 r})^n \\ &\geq \exp(-n e^{-c_3 r})) .\end{aligned}$$

Thus, for $n < C' e^{c' r}$, we have that this probability is greater than $7/8$. □

Hence, it is possible to learn the subspace even in the regime where it is informationally impossible to learn local solutions $w_i^\star$. However, this requires $n \gg d e^{cr}$ agents in total. The proof of this result differs drastically from that of Equation (9.4) and theorem 9.2 or proofs related such as that of [BKF22, RT11]. Indeed, with only one sample per agent, concentration results on the quantities considered (empirical losses mainly) will necessarily fail if we consider all possible local heads. Our approach thus only considers shared representations U: we compute the probability that a given representation is *admissible*, and extend this to all possible representations.

Note that results on algorithms such as Method of Moments [DFHT22, TJJ21] can be applied in the $m < r$ case. As mentioned in the introduction, since these methods are only specific to linear regression with quadratic losses, we do not consider them. Hence, even though such a setting is considered in this section, its purpose is to provide result for the low-rank estimator (Equation (9.5)) in this $m = 1$ setting. Our result shows that the low-rank estimator can indeed learn the representation, provided a number of tasks that scale exponentially large with r. The following proposition however suggests that this is necessary for our low rank estimator.

Proposition 9.4.2. *There exist constants $c', C' > 0$ such that if $n \leq C' e^{c' r}$ with probability $7/8$ there exist solutions $\hat{W}$ of the optimization problem Equation (9.4) such that the orthonormal projection $\hat{U}$ on the image of $\hat{W}$ are* orthogonal *to $U_\star$* i.e. $\frac{1}{r}\text{dist}^2_F(\hat{U}, U_\star) = 1$.

9.5. Convex relaxation via nuclear norm constraint

So far, except for the few-shot learning objective (9.6), the optimization problems considered to obtain the shared representation are the minimization of convex functions over *non-convex* constraint sets: low-rank matrices or clustered matrices. Despite the existence of heuristics to approximate solutions of these non-convex problems that are efficient in practice, there is no algorithm yet that can provably approximate solutions of Equations (9.4) and (9.5) in polynomial time. The objective of this section is thus to provide a convex relaxation of Equation (9.4) and prove generalization error bounds for the obtained relaxed estimator. Let, for some $\kappa > 0$ that satisfies $\kappa \geq \frac{\lambda_1(W^\star)}{\lambda_r(W^\star)}$ for $\lambda_1(W^\star)$ and $\lambda_r(W^\star)$ largest and $r-$largest singular values of $W^\star$:

$$\hat{W} = \arg\min \left\{ \mathcal{L}(W) , \quad W \in \mathbb{R}^{d \times n} , \quad \sup_i \|w_i\| \leq B , \quad \|W\|_* \leq \kappa B \sqrt{nr} \right\} , \tag{9.7}$$

and let $\hat{W}_{\text{SVD}}$ be the top-s SVD of $\hat{W}$ (the best rank s approximation of $\hat{W}$ in Frobenius norm), for some $s \in [\min(d, n)]$ to be determined. As opposed to the previous estimators, the obtained optimization problem consists of a convex objective minimized over a convex set. Importantly, notice that $W^\star$ lies in the optimization set $\mathcal{W}$. Two algorithms of choice to compute $\hat{W}$ are: **(i)** projected gradient descent (*a.k.a.* soft thresholding algorithm) onto the set $\mathcal{W}$, that provably leads to recover $\hat{W}$ since both objectives and constraint sets are convex. This involves projections on the ball $\{W : \|W\|_* = 1\}$, that have the algorithmic cost of

$\mathcal{O}(\min(n^2d, nd^2)$ (doing an SVD of matrices in $\mathbb{R}^{d\times n}$) plus that of ℓ^1-projections [DSSSC08], and **(ii)** Frank-Wolfe algorithm, that avoids such projections [Jag13], by iteratively performing line-searches along the direction of gradients, leading to optimizing linear functions over extreme-points of $\mathcal{W}$. The following results is derived from Theorem 9.5 that studies the relaxed estimator (9.7) under *ad-hoc* assumptions, as previously done in Lemma 9.3.1 for general estimators.

Theorem 9.4. *Assume that the assumptions of Theorem 9.1 hold and that each f_i is μ-strongly convex. With probability $1 - 4e^{-n} - 4e^{-nmd} - e^{-md}$, if $s = \sqrt{r(d+n)}$, the relaxed estimator $\hat{W}$ satisfies:*

$$\frac{1}{n}\left\|\hat{W}_{\mathrm{SVD}} - W^\star\right\|_F^2 = \tilde{\mathcal{O}}\left(\frac{\kappa^2 B^2 L}{\mu}\sqrt{\frac{r}{d+n}} + \frac{B^2((1+\kappa^2)\sigma^2 + (1+\kappa)\sigma_\star^2)}{\mu}\frac{\sqrt{r(d+n)}}{m}\right).$$

In this rate, the last term dominates, and the sample efficiency appears to be worse than for the non-convex low-rank estimator (Theorem 9.1). Indeed, for the right-hand side to vanish, we require $n \gg d$, which was also the case for the low-rank estimator, but we also need that $m \gg \sqrt{rd}$, as opposed to $m \gg r$ previously: this appears to be the cost of our relaxation. Note that similar rates were obtained by [BKF22] for linear quadratic regression. However, this $m \gg \sqrt{rd}$ condition, although being worse than $m \gg r$, is still much milder than the *no-collaboration* rate (Section 9.2.3), that requires $m \gg d$. In fact, the condition $m \gg \sqrt{rd}$ is exactly the geometric mean of the *no-collaboration* rate $m \gg d$ and the idealized *low-rank* rate $m \gg r$: it is a trade-off between statistical and computational efficiency.

Conclusion

In this chapter, we provided extensions to a well-studied problem: learning low-rank subspaces in multi-task or meta learning. While all existing works considered linear quadratic models, we relaxed this assumption to a more general convex setting, allowing to grasp other important problems, such as classification problems. We provided the first bounds for learning the subspace for the low rank estimator in the 1 sample per task setting, and highlighted the fact that few samples per task form a challenging statistical problem. Finally, we studied a nuclear norm relaxation for the rank constraint, and proved that this relaxation perfectly interpolates between the number of samples required by optimal rank-regularized estimator and the no-collaboration setting via their geometric mean.

Appendix of Chapter 9

9.A. Proof of Theorems 9.1 and 9.2

Proof of Theorem 9.1

Proof. Let $\mathcal{W} = \left\{W \in \mathbb{R}^{d\times n} \,|\, \operatorname{rank}(W) \leq r, \max_{i\in[n]} \|w_i\| \leq B\right\}$. We begin with the two following lemmas, from which Theorem 9.1 is directly derived.

Lemma 9.A.1. *With probability* $1 - 2e^{r(d+n)} - 2e^{nmd}$*, we have that, for all* $\Delta \in \mathcal{W}$*,*

$$|\langle \nabla\mathcal{L}(W^\star), \Delta\rangle| \leq C_1\sigma_\star^2 \frac{r(d+n)\ln\left(\frac{\sigma_\star^2 nmd}{r(d+n)}\right)}{nm} \sup_{i\in[n]} \|\Delta_i\|^2 + C_1\sigma_\star^2 \sqrt{\frac{r(d+n)\ln\left(\frac{\sigma_\star^2 nmd}{r(d+n)}\right)}{nm} \frac{1}{n}\|\Delta\|_F^2 \sup_{i\in[n]} \|\Delta_i\|^2}\,.$$

Proof of Lemma 9.A.1. Let $\mathcal{W}' = \left\{W \in \mathbb{R}^{d\times n} \,|\, \operatorname{rank}(W) \leq r, \max_{i\in[n]} \|w_i\| = B\right\}$ (by homogeneity of the result, we can impose $\max_{i\in[n]} \|w_i\| = B$). Let $\mathcal{W}_\varepsilon$ be an $\varepsilon-$net of $\mathcal{W}'$: we know that we can have $|\mathcal{W}_\varepsilon| \leq e^{cr(d+n)\ln(B/\varepsilon)}$. Using Lemma 9.3.2,

$$\mathbb{P}\left(|\langle \nabla\mathcal{L}(W^\star), \Delta\rangle| \geq C_1'\sigma_\star^2 t \sup_{i\in[n]} \|\Delta_i\|^2 + C_1'\sigma_\star^2\sqrt{\frac{t}{n}\|\Delta\|_F^2 \sup_{i\in[n]} \|\Delta_i\|^2}\right) \leq 2\exp\left(-nmt\right)\,.$$

Thus,

$$\mathbb{P}\left(\exists\Delta \in \mathcal{W}_\varepsilon\,,\, |\langle \nabla\mathcal{L}(W^\star), \Delta\rangle| \geq C_1'\sigma_\star^2 t \sup_{i\in[n]} \|\Delta_i\|^2 + C_1'\sigma_\star^2\sqrt{\frac{t}{n}\|\Delta\|_F^2 \sup_{i\in[n]} \|\Delta_i\|^2}\right) \leq 2\exp\left(-nmt + cr(d+n)\ln(B/\varepsilon)\right)\,.$$

Then, $|\langle \nabla\mathcal{L}(W^\star), \Delta\rangle|$ is $\sup_{i,j} \|\nabla F_i(w_i^\star, \xi_{ij})\|$-Lipschitz in Δ. Since $\nabla F_i(w_i^\star, \xi_{ij})$ are independent $\sigma_\star^2$ (multivariate) subexponential random variables. Thus, with probability $1 - 2e^{-nmd}$, we have $\sup_{i,j} \|\nabla F_i(w_i^\star, \xi_{ij})\| \leq \sigma_\star^2 d\ln(nm)$. Hence, using that $\mathcal{W}_\varepsilon$ is and $\varepsilon-$net of $\mathcal{W}$,

$$\mathbb{P}\left(\exists\Delta \in \mathcal{W}'\,,\, |\langle \nabla\mathcal{L}(W^\star), \Delta\rangle| \geq \varepsilon\sigma_\star^2 d\ln(nm) + C_1'\sigma_\star^2 t \sup_{i\in[n]} \|\Delta_i\|^2 + C_1'\sigma_\star^2\sqrt{\frac{t}{n}\|\Delta\|_F^2 \sup_{i\in[n]} \|\Delta_i\|^2}\right) \leq 2\exp\left(-nmt + cr(d+n)\ln(B/\varepsilon) + 2e^{-nmd}\right)\,.$$

We conclude by taking $t = \frac{cr(d+n)(1+\ln(B/\varepsilon))}{nm}$ for $\varepsilon = \frac{1}{\sigma_\star^2 d\ln(nm)}\frac{Br(d+n)}{nm}$. □

Lemma 9.A.2. *With probability* $1 - 2e^{r(d+n)} - 2e^{nmd}$*, we have that, for all* $(W, W') \in \mathcal{W}^2$*,*

$$\left|D_{f-\mathcal{L}}(W, W')\right|$$

$$\leq C_2\sigma^2 \sup_{i\in[n]} \left\|w_i - w_i'\right\|^2 \ln\left(\frac{nm(LB+\sigma_\star^2 nmd)}{\sigma^2 Br(d+n)}\right) \frac{r(d+n)}{nm} \ln\left(\frac{HBnm}{r(d+n)}\right)$$
$$+ C_2\sigma^2 \sqrt{\frac{1}{n}\|W-W'\|_F^2 \sup_{i\in[n]} \|w_i - w_i'\|^2 \ln\left(\frac{nm(LB+\sigma_\star^2 nmd)}{\sigma^2 Br(d+n)}\right) \frac{r(d+n)}{nm} \ln\left(\frac{HBnm}{r(d+n)}\right)}\,.$$

Proof of Lemma 9.A.2. Let $\mathcal{W}' = \{(W,W') \in \mathcal{W}, \sup_i \|w_i - w_i'\| = B\}$. Using Lemma 9.3.3,

$$\mathbb{P}\left[\left|D_{f-\mathcal{L}}(W,W')\right| > C_2\sigma^2 \sup_{i\in[n]} \left\|w_i - w_i'\right\|^2 t \ln\left(\frac{H\sup_{i\in[n]}\|w_i - w_i'\|}{t}\right)\right.$$
$$\left. + C_2\sigma^2\sqrt{\frac{\sigma^4}{n}\|W-W'\|_F^2 \sup_{i\in[n]} \|w_i - w_i'\|^2 t \ln\left(\frac{H\sup_{i\in[n]}\|w_i - w_i'\|}{t}\right)}\right]$$
$$\leq 2\exp\left(-nmt\right)\,.$$

Hence,

$$\mathbb{P}\left[\exists (W,W') \in \mathcal{W}_\varepsilon^2\,,\, \left|D_{f-\mathcal{L}}(W,W')\right| > C_2\sigma^2 \sup_{i\in[n]} \left\|w_i - w_i'\right\|^2 t \ln\left(\frac{H\sup_{i\in[n]}\|w_i - w_i'\|}{t}\right)\right.$$
$$\left. + C_2\sigma^2\sqrt{\frac{\sigma^4}{n}\|W-W'\|_F^2 \sup_{i\in[n]} \|w_i - w_i'\|^2 t \ln\left(\frac{H\sup_{i\in[n]}\|w_i - w_i'\|}{t}\right)}\right]$$
$$\leq 2\exp\left(-nmt + 2cr(d+n)\ln(B/\varepsilon)\right)\,.$$

$|D_{f-\mathcal{L}}(W,W')|$ is $(LB + \sup_{i,j}\|\nabla F_i(w_i^\star, \xi_{ij})\|)-$Lipschitz in W and in W', so that, since $\mathcal{W}_\varepsilon$ is an $\varepsilon-$net of $\mathcal{W}$,

$$\mathbb{P}\left[\exists (W,W') \in \mathcal{W}'\,,\, \left|D_{f-\mathcal{L}}(W,W')\right| > C_2\sigma^2 \sup_{i\in[n]} \left\|w_i - w_i'\right\|^2 t \ln\left(\frac{H\sup_{i\in[n]}\|w_i - w_i'\|}{t}\right)\right.$$
$$+ 2\varepsilon(LB + \sup_{i,j}\|\nabla F_i(w_i^\star, \xi_{ij})\|)$$
$$\left. + C_2\sigma^2\sqrt{\frac{\sigma^4}{n}\|W-W'\|_F^2 \sup_{i\in[n]} \|w_i - w_i'\|^2 t \ln\left(\frac{H\sup_{i\in[n]}\|w_i - w_i'\|}{t}\right)}\right]$$
$$\leq 2\exp\left(-nmt + 2cr(d+n)\ln(B/\varepsilon)\right)\,.$$

Then, since with probability $1-2e^{-nmd}$ we have $\sup_{i,j}\|\nabla F_i(w_i^\star,\xi_{ij})\| \leq \sigma_\star^2 nmd$, taking $t = \frac{(2c\ln(B/\varepsilon)+1)r(d+n)}{nm}$ and $\varepsilon = \frac{\sigma^2 B^2 r(d+n)}{nm}\frac{1}{LB+\sup_{i,j}\|\nabla F_i(w_i^\star,\xi_{ij})\|}$, we have ou result. □

Finally, we conclude the proof using Lemma 9.3.1. □

Proof of Theorem 9.2

Proof. Same as Theorem 9.1, but the $\varepsilon-$net cardinality scales as $r^n \times e^{crd\ln(1/\varepsilon)}$. □

9.B. Proof of Theorem 9.4

9.B.1. A general result

Theorem 9.5. *Assume that each f_i is μ-strongly convex and that for some constants $\alpha, \beta > 0$, for all $W, W', \Delta \in \mathbb{R}^{d\times n}$ of rank at most $s \geq 2r$, we have*

$$|\langle \nabla \mathcal{L}(W^\star), \Delta\rangle| \leq \alpha\sigma_\star^2 \sup_{i\in[n]} \|\Delta_i\|_2^2 + \sigma_\star^2 \sqrt{\frac{\alpha}{n}\|\Delta\|_F^2 \sup_{i\in[n]} \|\Delta_i\|_2^2}\,,$$

and

$$\left|D_{f-\mathcal{L}}(W, W')\right| \leq \beta\sigma^2 \sup_{i\in[n]} \left\|w_i - w_i'\right\|_2^2 + \sigma^2 \sqrt{\frac{\beta}{n}\|W - W'\|_F^2 \sup_{i\in[n]} \|w_i - w_i'\|_2^2}\,.$$

Assume furthermore that $\mathcal{L}$ satisfies $D_{\mathcal{L}}(W, W') \leq M\frac{\|W-W'\|_F^2}{n}$ for all $W, W' \in \mathbb{R}^{d\times n}$. Then, We have:

$$\begin{aligned}\frac{1}{n}\left\|\hat{W}_{\mathrm{SVD}} - W^\star\right\|_F^2 \leq \frac{2M\kappa^2 r}{\mu s} + \frac{4B}{\mu}\left[\sigma_\star^2\left(\frac{\alpha}{s} + \sqrt{\frac{\alpha}{ns}}\right)\right]\kappa\sqrt{nr} \\ + \frac{16B^2\beta\sigma^2}{\mu} + \frac{8B^2\sigma_\star^2}{\mu}\left(\alpha + \sqrt{\alpha}\right)\,.\end{aligned}$$

Proof. Importantly, notice that $W^\star$ lies in the optimization set

$$\mathcal{W} = \left\{W \in \mathbb{R}^{d\times n} \,\middle|\, \sup_i \|w_i\| \leq B\,, \quad \|W\|_* \leq \kappa B\sqrt{nr}\right\}.$$

Indeed, by assumption we first have that $\sup_i \|w_i^\star\| \leq B$, and $\|W^\star\|_* \leq r\lambda_{\max}(W) \leq \kappa\sqrt{rn}$. The scaling of κ then comes from the fact that if $W^\star$ is of rank r and is well-conditioned (its largest and $r-$largest singular values $\lambda_1(W^\star)$ and $\lambda_r(W^\star)$ have a bounded ratio), we have *(i)* $r\lambda_r(W^\star)^2\|W^\star\|_F^2 = \sum_i \|w_i^\star\|_2^2 \leq nB^2$ leading to λ and *(ii)* $\|W^\star\|_* \leq r\lambda_1(W^\star)$, so that if $\kappa \geq \frac{\lambda_1(W^\star)}{\lambda_r(W^\star)}$, we have $\|W^\star\|_* \leq r\lambda_1(W^\star) \leq \kappa r\lambda_r(W^\star) \leq \kappa B\sqrt{nr}$. We have:

$$D_{\mathcal{L}}(\hat{W}, W^\star) \leq \langle \nabla\hat{\mathcal{L}}(W^\star), W^\star - \hat{W}\rangle\,.$$

We first bound the noise term $\langle \nabla\hat{\mathcal{L}}(W^\star), W^\star - \hat{W}\rangle$. Let $\Delta = W^\star - \hat{W}$ and $\Delta = \sum_{k=1}^K$ for $K \leq \lceil\frac{\min(n,d)}{s}\rceil$ be its s-shelling decomposition, defined as follows.

Definition 9.B.1. *Let $W \in \mathbb{R}^{p\times q}$ and $s \in \mathbb{N}^*$. Let $W = \sum_{k=1}^{\min(p,q)} \sigma_k u_v v_k^\top$ be the singular value decomposition of W, where $\sigma_1 \geq \sigma_2 \geq \ldots \geq 0$. The top-$s$-SVD of W, abbreviated as $\mathrm{SVD}_s(W)$ is defined as $\mathrm{SVD}_s(W) = \sum_{k=1}^s \sigma_k u_v v_k^\top$.*

Definition 9.B.2. *Let $W \in \mathbb{R}^{p\times q}$ and $s \in \mathbb{N}^*$. Let $W = \sum_{k\geq 1} \Delta^{(k)}$ be the "shelling decomposition" of W, where the sequence of matrices $(\Delta^{(k)})_{k\geq 1}$ is defined as:*

1. *$\Delta^{(1)} = \mathrm{SVD}_s(W)$;*
2. *Recursively, for $k \geq 2$, $\Delta^{(k)} = \mathrm{SVD}_s(W - \sum_{\ell<k} \Delta^{(\ell)})$.*

Note that $\Delta^{(k)} = 0$ for $k \geq \lceil\min(p,q)/s\rceil$.

Lemma 9.B.1. *For any $W \in \mathbb{R}^{p\times q}$ and $s \in \mathbb{N}^*$, writing $W = \sum_{k\geq 1} \Delta^{(k)}$ the shelling decomposition of W, we have:*

$$\sum_{k\geq 2} \sup_{i\in[q]} \left\|\Delta_i^{(k)}\right\|_2 \leq \frac{\|W\|_*}{s}\,,$$

and

$$\sum_{k\geq 2} \left\|\Delta^{(k)}\right\|_F \leq \frac{\|W\|_*}{\sqrt{s}}\,.$$

We thus have:

$$\begin{aligned}
\langle \nabla\hat{\mathcal{L}}(W^\star), W^\star - \hat{W}\rangle &\leq \sum_{k\geq 1}\langle \nabla\hat{\mathcal{L}}(W^\star), \Delta^k\rangle \\
&\leq \sum_{k\geq 1} \alpha\sigma_\star^2 \sup_{i\in[n]}\|\Delta_i\|_2^2 + \sigma_\star^2\sqrt{\frac{\alpha}{n}\|\Delta\|_F^2 \sup_{i\in[n]}\|\Delta_i\|_2^2} \\
&\leq 4B^2\left(\alpha\sigma_\star^2 + \sigma_\star^2\sqrt{\alpha}\right) + \sum_{k\geq 2}\alpha\sigma_\star^2 \sup_{i\in[n]}\|\Delta_i\|_2^2 + \sigma_\star^2\sqrt{\frac{\alpha}{n}\|\Delta\|_F^2 \sup_{i\in[n]}\|\Delta_i\|_2^2} \\
&\leq 4B^2\sigma_\star^2\left(\alpha + \sqrt{\alpha}\right) + \sigma_\star^2\|\Delta\|_*\left(\frac{\alpha}{s} + B\sqrt{\frac{\alpha}{ns}}\right) \\
&= \varepsilon_0\,,
\end{aligned}$$

where ε_0 is defined by this last equality. Then, we have that $\|\Delta\|_* \leq 2\kappa\sqrt{nr}$, and thus

$$\varepsilon_0 \leq 4B^2\sigma_\star^2\left(\alpha + \sqrt{\alpha}\right) + 2\kappa\sqrt{nr}\sigma_\star^2\left(\frac{\alpha}{s} + B\sqrt{\frac{\alpha}{ns}}\right)\,.$$

We now need to lower bound $D_{\mathcal{L}}(\hat{W}, W^\star)$. Let $\hat{W} = \sum_{k=1}^K \hat{\Delta}^k$ be its s-shelling decomposition, and let $\tilde{\Delta}^k = \hat{\Delta}^k$ for $k \geq 2$ and $\tilde{\Delta}^1 = \hat{\Delta}^1 - W^\star$. Let $\tilde{\Delta} = \hat{W} - W^\star = \sum_k \tilde{\Delta}^k$. We have:

$$\begin{aligned}
D_{\mathcal{L}}(\hat{W}, W^\star) &= D_{\mathcal{L}}(W^\star + \tilde{\Delta}, W^\star) \\
&= D_{\mathcal{L}}(W^\star + \sum_{k=1}^K \tilde{\Delta}^k, W^\star) \\
&= D_{\mathcal{L}}(W^\star + \tilde{\Delta}^1 + \sum_{k=2}^K \tilde{\Delta}^k, W^\star) \\
&\geq D_{\mathcal{L}}(W^\star + \tilde{\Delta}^1 + \sum_{k=2}^K \tilde{\Delta}^k, W^\star) \\
&\geq 2D_{\mathcal{L}}(W^\star + \frac{1}{2}\tilde{\Delta}^1, W^\star) - D_{\mathcal{L}}(W^\star - \sum_{k=2}^K \tilde{\Delta}^k, W^\star) \\
&\geq 2D_{\mathcal{L}}(W^\star + \frac{1}{2}\tilde{\Delta}^1, W^\star) - \frac{1}{K-1}\sum_{k=2}^K D_{\mathcal{L}}(W^\star - (K-1)\tilde{\Delta}^k, W^\star)\,,
\end{aligned}$$

where we used twice the convexity of $D_{\mathcal{L}}(W^\star + \cdot, W^\star)$.

Then,

$$\begin{aligned}
D_{\mathcal{L}}(W^\star + \frac{1}{2}\tilde{\Delta}^1, W^\star) &= D_f(W^\star + \frac{1}{2}\tilde{\Delta}^1, W^\star) + D_{\mathcal{L}-f}(W^\star + \frac{1}{2}\tilde{\Delta}^1, W^\star) \\
&\geq D_f(W^\star + \frac{1}{2}\tilde{\Delta}^1, W^\star) - \sigma^2\left(\beta\sup_i\left\|\tilde{\Delta}_i^1\right\|_2^2 + \sqrt{\frac{\beta}{n}\left\|\tilde{\Delta}^1\right\|_F^2 \sup_i\left\|\tilde{\Delta}_i^1\right\|_2^2}\right)
\end{aligned}$$

$$\geq D_f(W^\star + \frac{1}{2}\tilde{\Delta}^1, W^\star) - \sigma^2\left(4B^2\beta + 2B\sqrt{\frac{\beta}{n}\left\|\tilde{\Delta}^1\right\|_F^2}\right),$$

and using the property with constant M:

$$\begin{aligned} D_{\mathcal{L}}(W^\star - \sum_{k=2}^{K}\tilde{\Delta}^k, W^\star) &\leq \frac{M}{n}\left\|\sum_{k=2}^{K}\tilde{\Delta}^k\right\|_F^2 \\ &\leq \frac{M\|W^\star\|_*^2}{ns} \\ &\leq \frac{M\kappa^2 r}{s} \end{aligned}$$

Now, wrapping everything up,

$$\begin{aligned} 2(f(W^\star + \tilde{\Delta}^1/2) - f(W^\star)) \leq \frac{M\kappa^2 r}{s} &+ 2\left[\sigma_\star^2\left(\frac{\alpha}{s} + B\sqrt{\frac{\alpha}{ns}}\right)\right]\kappa\sqrt{nr} \\ &+ \sigma^2\left(4B^2\beta + 2B\sqrt{\frac{\beta}{n}\left\|\tilde{\Delta}^1\right\|_F^2}\right) + 4B^2\sigma_\star^2\left(\alpha + \sqrt{\alpha}\right). \end{aligned}$$

Then, if each f_i is μ strongly convex, f is μ/n strongly convex and:

$$\begin{aligned} \frac{\mu}{n}\left\|\tilde{\Delta}^1\right\|_F^2 \leq \frac{M\kappa^2 r}{s} &+ 2\left[\sigma_\star^2\left(\frac{\alpha}{s} + B\sqrt{\frac{\alpha}{ns}}\right)\right]\kappa\sqrt{nr} \\ &+ \sigma^2\left(4B^2\beta + 2B\sqrt{\frac{\beta}{n}\left\|\tilde{\Delta}^1\right\|_F^2}\right) + 4B^2\sigma_\star^2\left(\alpha + \sqrt{\alpha}\right). \end{aligned}$$

Thus,

$$\begin{aligned} \frac{1}{n}\left\|\hat{W}_{\mathrm{SVD}} - W^\star\right\|_F^2 \leq \frac{2M\kappa^2 r}{\mu s} &+ \frac{4}{\mu}\left[\sigma_\star^2\left(\frac{\alpha}{s} + B\sqrt{\frac{\alpha}{ns}}\right)\right]\kappa\sqrt{nr} \\ &+ \frac{16B^2\beta\sigma^2}{\mu} + \frac{8B^2\sigma_\star^2}{\mu}\left(\alpha + \sqrt{\alpha}\right). \end{aligned}$$

□

9.B.2. Proof of Theorem 9.4

The first two assumptions of Theorem 9.5 are satisfied with high probability for $\alpha, \beta = \tilde{\mathcal{O}}(\frac{s(d+n)}{nm})$, as proved in the proof of Theorem 9.1. We now prove that the third and last assumption is satisfied for $M = L + \tilde{\mathcal{O}}(\sigma^2 d/m)$ with high probabiliy.

Proof.

Lemma 9.B.2. *With probability* $1 - e^{-cmd}$, *for all* $i \in [n]$, *for all* $w, w' \in \mathbb{R}^d$,

$$D_{\mathcal{L}_i}(w, w') \leq \left(L + c'\sigma^2\frac{d}{m}\ln(dHB^2/\sigma^2)\right)\left\|w - w'\right\|^2$$

Proof. We have, for some $v \in [w, w']$:

$$D_{\mathcal{L}_i}(w, w') = \frac{1}{2}\left\|w - w'\right\|_{\nabla^2\mathcal{L}_i(v)}^2$$

$$= \frac{1}{m}\sum_j (w-w')^\top \nabla^2 F_i(v,\xi_{ij})(w-w')\,.$$

Assume that $\|w-w'\|=1$. Now, all $(w-w')^\top \nabla^2 F_i(v,\xi_{ij})(w-w')$ are σ^2-subexponential and independent, of mean $(w-w')^\top \nabla^2 f_i(v)(w-w') \le L$, leading to:

$$\mathbb{P}\left(\sum_j (w-w')^\top \nabla^2 F_i(v,\xi_{ij})(w-w') \ge mL + \sigma^2 t\right) \le 2\exp\left(-c\min(\frac{t^2}{m}, t)\right)\,,$$

so that:

$$\mathbb{P}\left(\frac{1}{m}\sum_j (w-w')^\top \nabla^2 F_i(v,\xi_{ij})(w-w') \ge L + \sigma^2 t\right) \le 2\exp\left(-cm\min(t^2,t)\right)\,.$$

Using discretization arguments as before, both on $v\in[w,w']$ parameterized by some $\lambda\in[0,1]$ and on w,w', we get that, with probability $1-e^{-cmd}$, for all w,w' such that $\|w\|,\|w'\|\le B$,

$$D_{\mathcal{L}_i}(w,w') \le \left(L + c'\sigma^2\frac{d}{m}\ln(dHB^2/\sigma^2)\right)\|w-w'\|^2$$

□

□

Part III

Additional contributions of this thesis

Chapter 10

Concentration of random tensors and of Hessians, applications to preconditionning

Dimension is an inherent bottleneck to some modern learning tasks, where optimization methods suffer from the size of the data. In this chapter, we study non-isotropic distributions of data and develop tools that aim at reducing these dimensional costs by a dependency on an effective dimension rather than the ambient one. Based on non-asymptotic estimates of the metric entropy of ellipsoids -that prove to generalize to infinite dimensions- and on a chaining argument, our uniform concentration bounds involve an *effective dimension* instead of the global dimension, improving over existing results. We show the importance of taking advantage of non-isotropic properties in learning problems where uniform concentration of Hessians is required. In particular, we improve state-of-the-art results in statistical preconditioning for communication-efficient distributed optimization, where we additionally provide stochastic updates in the Bregman framework.

Other applications such as randomized smoothing or robustness of neural networks are also present in [EM21], the paper from which this chapter is based. We refer the reader to the associated paper for proofs, that are not included in the manuscript.

10.1. Introduction

The sum of *i.i.d.* symmetric random tensors of order 2 and rank 1 (*i.e.* symmetric random matrices of rank 1) is studied in probability and statistics both for theoretical and practical interests, the most classical application being covariance estimation. The empirical mean of such matrices follows the Wishart distribution [Wis28, Uhl94]. [MP67] proved the convergence in law of their spectrum when the number of observations and the dimension are of the same order. Machine Learning applications however require non-asymptotic properties, such as concentration bounds for a potentially large finite number of observations and finite dimension [Tro11, T$^+$15, DGJ17, Min17], to control the eigenvalues of sums of independent matrices, namely:

$$\left\| \frac{1}{n} \sum_{i=1}^{n} a_i a_i^\top - \mathbb{E}\left[aa^\top\right] \right\|_{\text{op}} = \sup_{\|x\| \leq 1} \frac{1}{n} \sum_{i=1}^{n} x^\top \left(a_i a_i^\top - \mathbb{E}\left[aa^\top\right] \right) x \tag{10.1}$$

for $a, a_1, ..., a_n$ *i.i.d.* random variables in $\mathbb{R}^d$.

10.1.1. Contributions and overview of this chapter

Our main contribution consists in new tools for the control of quantities generalizing (10.1). More precisely, for $r \geq 2$, $f_1, ..., f_r$ Lipschitz functions on $\mathbb{R}$, $a, a_1, ..., a_n$ *i.i.d.* random variables in $\mathbb{R}^d$, and $\mathcal{B}$ the d-dimensional unit ball, we derive in Section 10.2 concentration

bounds on:

$$\sup_{x_1,...,x_r \in \mathcal{B}} \left\{ \frac{1}{n} \sum_{i \in [n]} \left(\prod_{k=1}^{r} f_k(a_i^\top x_k) - \mathbb{E}\left[\prod_{k=1}^{r} f_k(a^\top x_k) \right] \right) \right\}. \tag{10.2}$$

We thereby extend previous results in three directions. *i)* Matrices are tensors of order 2, which we generalize by treating symmetric random tensors of rank 1 and order $r \geq 2$ (Section 10.2.3). *ii)* We consider non linear functions f_i of scalar products $\langle a_i, x \rangle$, motivated by Empirical Risk Minimization. (10.2) can thus be seen as the uniform maximum deviation of a symmetric random tensor of order r and rank 1, with non-linearities $f_1, ..., f_r$. *iii)* Finally, by observing that data are usually distributed in a non-isotropic way (the *MNIST* dataset lies in a 712 dimensional space, yet its empirical covariance matrix is of effective dimension less than 3 for instance), we generalize classical isotropic assumptions on random variables a_i by introducing a non-isotropic counterpart:

Definition 10.1.1 (Σ-Subgaussian Random Vector)**.** *A random variable a with values in $\mathbb{R}^d$ is Σ -subgaussian for $\Sigma \in \mathbb{R}^{d \times d}$ a positive-definite matrix if:*

$$\forall t > 0, \forall x \in \mathcal{B}, \mathbb{P}(|a^\top x| > t) \leq 2 \exp\left(-\frac{1}{2} \frac{t^2}{x^\top \Sigma x} \right). \tag{10.3}$$

A gaussian $\mathcal{N}(0, \Sigma)$ is for instance Σ-subgaussian. Note however that in the general case, Σ is not equal to the covariance matrix. The aim is then to derive concentration bounds on (10.2) (Section 10.2) that involve an *effective dimension* of Σ: a quantity smaller than the global dimension d, that reflects the non-isotropic repartition of the data:

Definition 10.1.2 (Effective Dimension $d_{\text{eff}}(r)$)**.** *Let $\Sigma \in \mathbb{R}^{d \times d}$ a symmetric positive semi-definite matrix of size $d \times d$, where $d \in \mathbb{N}^*$. Let $\sigma_1^2 \geq \sigma_2^2 \geq ... \geq \sigma_d^2 \geq 0$ denote its ordered eigenvalues. For any $r \in \mathbb{N}^*$, let $d_{\text{eff}}(r)$ be defined as follows:*

$$d_{\text{eff}}(r) := \sum_{i=1}^{d} \left(\frac{\sigma_i}{\sigma_1} \right)^{\frac{2}{r}} = \frac{\text{Tr}(\Sigma^{1/r})}{\left\| \Sigma^{1/r} \right\|_{\text{op}}}. \tag{10.4}$$

This notion generalizes intrinsic dimension *in [T+15] and* stable rank *in [Ver11b, Ver14], both obtained for $r = 1$.*

Chaining Argument and Metric Entropy of Ellipsoids: Control of (10.2) involves a chaining argument ([BLM13], Chapter 13). In the simplest version of chaining, in order to bound a random variable of the form $\sup_{t \in \mathcal{T}} X_t$, one discretizes the set of indices $\mathcal{T}$ and approximates the value $\sup_{t \in \mathcal{T}} X_t$ by a supremum taken over successively refined discretizations. To exploit the non-isotropic properties of Σ-subgaussian random variables, we apply chaining based on a covering of the unit ball $\mathcal{B}$ with ellipsoids. Our approach yields similarities with that of [Zho17], who uses chaining with ellipsoids for a different purpose (control of eigenvectors). In section 10.2.3, in the setting where $f_1 = ... = f_r = Id$, control of (10.2) reduces to controlling the operator norm of empirical tensors. This can be done using our bounds on the ε-entropy of ellipsoids, without the use of chaining.

In Section 10.3, we present results on the number of balls of fixed radius ε needed to cover an ellipsoid in dimension d. The logarithm of this quantity is often called the *ε-entropy* of an ellipsoid. [DPP04] studied the limit $d \to \infty$, while we provide non-asymptotic estimates. Furthermore, we extend these results to ellipsoids in infinite dimension, obtaining bounds on metric entropy in terms of power-law norm decay. We believe these technical results (both in finite and infinite dimension) to be of strong practical and theoretical interests: the bridge between covering numbers and suprema of random subgaussian processes is rather thin due to Dudley's inequality [Dud67]. Bounding metric entropy of ellipsoids is thus a step towards uniform bounds on more general random variables than the one we consider in (10.2).

10.1.2. Applications in Learning Problems and ERM

We show the relevance of our concentration bounds through the following applications.

Operator Norm Of Tensors Setting $f_1 = ... = f_r = Id$ yields the operator norm of the empirical tensor $\frac{1}{n}\sum_i a_i^{\otimes r} - \mathbb{E}a^{\otimes r}$ in (10.2). In Section 10.2.3 we derive precise large deviation bounds on such tensors involving the effective dimension $d_{\text{eff}}(r)$, improving on previous works [BLN20, PVZ17] which depended on the global dimension. Optimal concentration inequalities on rank 1 symmetric tensors (*i.e.* of the form $a^{\otimes r}$) are not known. We refer the interested reader to [Ver20] for the study of rank 1 tensors of the form $a_1 \otimes ... \otimes a_r$ where $a_1, ..., a_r$ are *i.i.d.* random variables, a different problem than ours. In [EM21], we apply these bounds to the study of the Lipschitz constant of two-layered neural networks with polynomial activation, elaborating on the results in [BLN20].

Concentration of Hessians and Statistical Preconditioning For ℓ a twice differentiable function on $\mathbb{R}$ and Hessian-Lipschitz, let $f(x) = \frac{1}{n}\sum_{i=1}^n \ell(a_i^\top x)$. Then,

$$\nabla^2 f(x) = \frac{1}{n}\sum_{i=1}^n \ell''(a_i^\top x) a_i a_i^\top ,$$

and setting $r = 3, f_1 = \ell'', f_2 = f_3 = Id$ in (10.2) yields $\sup_{x\in\mathcal{B}} \left\|\nabla^2 f(x) - \mathbb{E}[\nabla^2 f(x)]\right\|_{\text{op}}$. Controlling such quantities is relevant in optimization when studying functions that have an empirical risk structure. Methods such as statistical preconditioning [SSZ14] take advantage of the *i.i.d.* structure of the observations, as we illustrate in Section 10.4. Our results improve on the state of the state-of-the-art [HXB+20], establishing guarantees based on $d_{\text{eff}}(r)$ rather than d.

Randomized Smoothing We also present applications of our results to non-isotropic randomized smoothing in [EM21].

Organization of the chapter We first present our 3 main uniform concentration bounds in Section 10.2: control of (10.2) and of the same quantity but un-centered (both using chaining), and a more precise control of (10.2) in the case where $f_1 = ... = f_r = Id$ (control of empirical mean of symmetric random tensors of rank 1 ann order p). In section 10.3, we provide bounds on the metric entropy of ellipsoids in terms of effective dimension. We also investigate the case of infinite dimension with the notion of spectral dimension. The last two sections present two applications of the results presented in Section 10.2. In Section 10.4, we apply Theorem 10.1 to control uniform deviation of Hessians, in order to prove that statistical preconditioning methods naturally adapt to the underlying effective dimension.

10.2. Main Theoretical Results

10.2.1. Concentration Bound With Centering

Theorem 10.1 (Concentration With Centering)**.** *Let $r \geq 2$ and $d, n \geq 1$ integers. Let $\Sigma \in \mathbb{R}^{d\times d}$ a positive-definite matrix and $a, a_1, ..., a_n$* i.i.d. *$\Sigma-$subgaussian random variables. Let $d_{\text{eff}}(s), s \in \mathbb{N}^*$ be defined as in (10.4). Let $f_1, ..., f_r$ be 1-Lipshitz continuous functions on $\mathbb{R}$ such that $f_i(0) = 0$ for $i \in [n]$. For all $k = 1, ..., r$, let $B_k > 0$ such that:*

$$\forall x \in \mathcal{B}, \forall i \in [n], |f_k(a_i^\top x)| \leq B_k \text{ almost surely.} \tag{10.5}$$

Let $B = B_1...B_k$. Define the following random variable:

$$Z := \sup_{x_1,...,x_r \in \mathcal{B}} \left\{ \frac{1}{n} \sum_{i \in [n]} \left(\prod_{k=1}^{r} f_k(a_i^\top x_k) - \mathbb{E}\left[\prod_{k=1}^{r} f_k(a^\top x_k) \right] \right) \right\}. \tag{10.6}$$

Then, for any $\lambda > 0$ and for some universal constant C_r, the following large-deviation bound holds:

$$\mathbb{P}\left(Z \geq C_r \sigma_1^r \left(\frac{1}{n} \frac{\lambda + d_{\text{eff}}(r) \ln(d)}{\left(\sigma_1^{-r} B\right)^{2/r-1}} + \frac{\sqrt{\lambda} + \sqrt{d_{\text{eff}}(1) \ln(d)}}{\sqrt{n}} \right) \right) \leq e^{-\lambda}. \tag{10.7}$$

10.2.2. Concentration Bound Without Centering

Theorem 10.2 (Concentration Without Centering)**.** *Let $r \geq 2$ and $d, n \geq 1$ integers. Let $\Sigma \in \mathbb{R}^{d \times d}$ a positive-definite matrix and $a, a_1, ..., a_n$ i.i.d. $\Sigma-$subgaussian random variables (10.3). Let $d_{\text{eff}}(s), s \in \mathbb{N}^*$ be defined as in (10.4). Let $f_1, ..., f_r$ be 1-Lipshitz continuous functions on $\mathbb{R}$ such that $f_i(0) = 0$ for $i \in [n]$. For all $k = 1, ..., r$, let $B_k > 0$ such that:*

$$\forall x \in \mathcal{B}, \forall i \in [n], |f_k(a_i^\top x)| \leq B_k \text{ almost surely.}$$

Let $B = B_1...B_k$. Define the following random variable:

$$Y := \sup_{x_1,...,x_r \in \mathcal{B}} \frac{1}{n} \sum_{i \in [n]} \prod_{k=1}^{r} f_k(a_i^\top x_k). \tag{10.8}$$

Then, for any $\lambda > 0$ and for some universal constant C_r, the following large-deviation bound holds:

$$\mathbb{P}\left(Y \geq \sigma_1^r C_r \left(1 + \frac{d_{\text{eff}}(r) \ln(d) + \lambda}{n} (\sigma_1^{-r} B)^{1-2/r} \right) \right) \leq e^{-\lambda}. \tag{10.9}$$

Remark 10.2.1. *Assumptions (10.5) in Theorem 10.1 can be replaced by high-probability bounds on the random variables $a_1, ..., a_n$ in the following way. If we denote $R = \sup_{i=1,...,n} \|a_i\|$, we always have $B_k \leq R$ and $B \leq R^r$ using Lipshitz continuity of functions f_k. Furthermore, a Chernoff bound gives with probability $1 - \delta$:*

$$R^2 \leq 4\sigma_1^2 (2 d_{\text{eff}}(1) + \ln(1/\delta) + \ln(n)),$$

yielding, with probability $1 - \delta$, where Z is defined in (10.6):

$$Z \leq C_r \sigma_1^r \left(\frac{\ln(\delta^{-1}) + d_{\text{eff}}(r) \ln(d)}{n} \left(d_{\text{eff}}(1) + \ln(\delta^{-1}) + \ln(n) \right)^{\frac{r}{2}-1} + \frac{\sqrt{\ln(\delta^{-1})} + \sqrt{d_{\text{eff}}(1) \ln(d)}}{\sqrt{n}} \right).$$

The same reasoning applies to Theorem 10.2 in the case without centering.

Remark 10.2.2. *Theorems 10.1 and 10.2 assume that the functions f_k are 1-Lipshitz and that the supremum is taken over $\mathcal{B}$ the centered unit ball. By considering L_k-Lipshitz functions and a ball $\mathcal{B}(x_0, \rho)$, one obtains the same bound, up to a factor $\rho L_1 ... L_r$.*

10.2.3. Concentration of Non-Isotropic Random Tensors

In this section, we provide a concentration bound on the empirical mean of symmetric random tensors of rank 1, involving an effective dimension. In [EM21], we exploit this result to derive some results on the robustness of two-layered neural networks with polynomial activations. Methods such as in [PVZ17, BLN20], which do not rely ellipsoids, cannot yield results as sharp as ours, as detailed in the full version paper [EM21].

Definition 10.2.1 (Tensor). *A tensor of order $p \in \mathbb{N}^*$ is an array $T = (T_{i_1,...,i_p})_{i_1,...,i_p \in [d]} \in \mathbb{R}^{dp}$. T is said to be of rank 1 if it can be written as:*

$$T = u_1 \otimes \cdots \otimes u_p$$

for some $u_1, ..., u_p \in \mathbb{R}^p$.
Scalar product between two tensors of same order p is defined as:

$$\langle T, S \rangle = \sum_{i_1,...,i_p} T_{i_1,...,i_p} S_{i_1,...,i_p}, \text{ giving the norm: } \|T\|^2 = \sum_{i_1,...,i_p} T^2_{i_1,...,i_p}.$$

We define the operator norm of a tensor as:

$$\|T\|_{\text{op}} = \sup_{\|x_1 \otimes ... \otimes x_p\| \leq 1} \langle T, x_1 \otimes ... \otimes x_p \rangle.$$

Definition 10.2.2 (Symmetric Random Tensor of Rank 1). *A symmetric random tensor of rank 1 and order p is a random tensor of the form:*

$$T = X^{\otimes p}, \tag{10.10}$$

where $X \in \mathbb{R}^d$ is a random variable. We say that T is Σ-subgaussian if X is a Σ-subgaussian random variable.

We wish to bound the operator norm of tensors of the form $T = \frac{1}{n}\sum_{i=1}^n T_i$, where $T_1, ..., T_n$ are *i.i.d.* subgaussian random tensors of rank 1 and order p, using a dependency in an effective dimension rather than the global one. We have:

$$\|T - \mathbb{E}T\|_{\text{op}} = \frac{1}{n} \sup_{x_1,...,x_p \in \mathcal{S}} \sum_{i=1}^n \left\{ \prod_{k=1}^p \langle a_i, x_k \rangle - \mathbb{E}\left[\prod_{k=1}^p \langle a_i, x_k \rangle \right] \right\}.$$

This quantity can be upper-bounded using chaining as in Theorem 10.1. However, using a simpler argument inspired by [BLN20] and our bounds on the metric entropy of ellipsoids, we have the following.

Theorem 10.3 (Non-Isotropic Concentration Bound on Random Tensors). *Let $T_1, ..., T_n$ be* i.i.d. *random tensors of order p, rank 1, symmetric and Σ-subgaussian. Let $T = \frac{1}{n}\sum_{i=1}^n T_i$. With probability $1 - \delta$ for any $\delta > 0$ and universal constant $C_p > 0$, we have:*

$$\|T - \mathbb{E}T\|_{\text{op}} \leq C_p \sigma_1^p \sqrt{\frac{\left(d_{\text{eff}} + \ln(d) + \ln(\delta^{-1})\right)^{p+1} \ln(n)^p}{n}}.$$

Equivalently, for any $\lambda > 0$:

$$\mathbb{P}\left(\|T - \mathbb{E}T\|_{\text{op}} \geq C_p \sigma_1^p \sqrt{\frac{\left(d_{\text{eff}} + \ln(d) + \lambda\right)^{p+1} \ln(n)^p}{n}} \right) \leq e^{-\lambda}.$$

This concentration bound is similar to those of [Zhi21, Ver11a, GR07], up to logarithmic factors we lose in our approach. Yet, even though our proof is quite straightforward, we obtain results similar to some using much more involved and sophisticated arguments. We next present the upper-bounds on the number of balls needed to cover an ellipsoid we use to prove the concentration bounds with chaining (Theorems 10.1 and 10.2) or without (Theorem 10.3). We believe these technical results to be of independent interest due to the strong link between metric entropy and uniform concentration bounds.

10.3. Covering Balls with Ellipsoids and Metric Entropy

10.3.1. Metric Entropy of an Ellipsoid

Definition 10.3.1 (Ellipsoid and ε-Entropy)**.** *Given a vector $b = (b_1, \dots, b_d)$ with $b_1 \geq \cdots \geq b_d > 0$, the ellipsoid E_b is defined as*

$$E_b = \left\{ x \in \mathbb{R}^d : \sum_{i \in [d]} \frac{x_i^2}{b_i^2} \leq 1 \right\}.$$

The ε-entropy $\mathcal{H}_\varepsilon(E_b)$ of ellipsoid E_b is the logarithm of the size of a minimal ε-covering (or ε-net in information theory terminology) of E_b. More formally:

$$\mathcal{H}_\varepsilon(E_b) = \ln \left(\min \left\{ |A| : A \subset \mathbb{R}^d, E_b \subset \bigcup_{x \in A} \mathcal{B}(x, \varepsilon) \right\} \right), \tag{10.11}$$

where $\mathcal{B}(x, \varepsilon)$ is the Euclidean ball of radius ε. The unit entropy is the ε-entropy for $\varepsilon = 1$.

Given an ellipsoid E_b, define the following quantities:

$$K_b = \sum_{i=1}^{m_b} \ln(b_i) \text{ and } m_b = \sum_{i \in [d]} \mathbb{1}_{b_i > 1}. \tag{10.12}$$

Provided that:

$$\ln(b_1) = o\left(\frac{K_b^2}{m_b \ln(d)} \right), \tag{10.13}$$

[DPP04] (Theorem 2 in their article) prove the following asymptotic equivalent of $\mathcal{H}_1(E_b)$ when $d \to \infty$:

$$\mathcal{H}_1(E_b) \sim K_b. \tag{10.14}$$

However, we need non-asymptotic bounds on $\mathcal{H}_1(E_b)$. Using techniques introduced in [DPP04], we thus establish Theorem 10.4, whose proof appears [EM21], together with an extension to ellipsoids in infinite dimension.

Theorem 10.4 (Unit Entropy of an Ellipsoid in Fixed Dimension)**.** *One has, for some universal constant $c > 0$, the following bound on the unit entropy of ellipsoid E_b:*

$$\mathcal{H}_1(E_b) \leq K_b + c \left[\ln(d) + \sqrt{\ln(b_1) m_b \ln(d)} \right].$$

This theorem gives the following corollary, bounding the number of ellipsoids required to cover the unit ball, directly linked with the number of balls required to cover an ellipsoid thanks to a linear transformation.

10.3.2. Coverings of the Unit Ball With Ellipsoids

Corollary 10.3.1. *Let $\varepsilon > 0$. Let random vector $a \in \mathbb{R}^d$ satisfy subgaussian tail assumption (10.3) for matrix Σ, with spectrum $\sigma_1^2 \geq \cdots \geq \sigma_d^2 > 0$. Then there exists a collection $\mathcal{N}_\varepsilon$ of vectors in $\mathcal{S}_1$ the unit sphere of $\mathbb{R}^d$ such that, for all $x \in \mathcal{S}_1$, there exists $y = \Pi_\varepsilon x \in \mathcal{N}_\varepsilon$ such that*

$$\|x - y\|_\Sigma^2 := (x - y)^\top \Sigma (x - y) \leq \varepsilon^2 \sigma_1^2, \tag{10.15}$$

and the covering $\mathcal{N}_\varepsilon$ verifies

$$\ln(|\mathcal{N}_\varepsilon|) \leq \mathcal{H}_\varepsilon := \sum_{i=1}^{m_\varepsilon} \ln\left(\frac{\sigma_i}{\varepsilon \sigma_1} \right) + c \left[\ln(d) + \sqrt{\ln(\varepsilon^{-1}) \ln(d) m_\varepsilon} \right], \tag{10.16}$$

where

$$m_\varepsilon = \sum_{i=1}^{d} \mathbb{1}_{\sigma_i > \varepsilon\sigma_1} \tag{10.17}$$

and c is some universal constant. Furthermore, we have:

$$\mathcal{H}_\varepsilon \leq \ln(\varepsilon^{-1}) + \frac{\min\left(d-1, \varepsilon^{-\frac{2}{r}}\frac{d_{\text{eff}}(r)-1}{e}\right)}{2/r} \ln\left(\max(e, \varepsilon^{-\frac{2}{r}}\frac{d_{\text{eff}}(r)-1}{d-1})\right) + c\left[\ln(d) + \sqrt{\ln(\varepsilon^{-1})\ln(d)m_\varepsilon}\right],$$

and

$$m_\varepsilon \leq 1 + (d_{\text{eff}}(r)-1)\varepsilon^{-\frac{2}{r}}.$$

This last bound on $\mathcal{H}_\varepsilon$ is a core technical lemma behind Theorems 10.1 and 10.2 . It is to be noted that $\mathcal{H}_\varepsilon$ is not linear in an effective dimension. Indeed, for $\varepsilon \leq C_r\left(\frac{d-1}{d_{\text{eff}}(r)-1}\right)^{r/2}$, our expression is linear in d. This difficulty is the non-asymptotic equivalent of [DPP04]'s assumption in (10.13).

10.3.3. Ellipsoids in Infinite Dimension

We here define ellipsoids in infinite dimension and upper-bound asymptotically their ε-entropy in terms of *spectral dimension*. Although not used in the applications described in the present article, uniform concentration of infinite-dimensional random vectors that satisfy an infinite-dimensional subgaussian property require results such as the one we provide below.

Let $\mathcal{V}$ be a separable real Hilbert space (*e.g.* $\mathbb{R}^{\mathbb{N}}$, $\ell^2([0,1])$).

Definition 10.3.2 (Ellipsoids in Hilbert Spaces)**.** *Let A a self-adjoint and semi-definite positive operator on $\mathcal{V}$* i.e. *such that $\forall (x,y) \in \mathcal{V}^2$, we have $\langle A(x), y\rangle = \langle x, A(y)\rangle \geq 0$. We define the ellipsoid $E_A \subset \mathcal{V}$ by:*

$$E_A = \left\{x \in \mathcal{V} : \left\|A^\dagger(x)\right\|^2 \leq 1\right\},$$

where $A^\dagger$ is the pseudo-inverse of A.

This notion generalizes Definition 10.3.1: taking $\mathcal{V} = \mathbb{R}^d$ and $A = \text{Diag}(b_1, ..., b_d)$, we have $E_A = E_b$. We next define the spectral dimension of an ellipsoid. We recall that if A is a self-adjoint and semi-definite positive operator on $\mathcal{V}$, there exists a Hilbert basis of eigenvectors of A, and the eigenvalues of A are non-negative.

Definition 10.3.3 (Spectral Dimension and Effective Dimension)**.** *Let E_A an ellipsoid in $\mathcal{V}$, where A is a self-adjoint and semi-definite positive operator. Assume that the eigenvalues of A can be ordered as a decreasing sequence $(b_i)_{i\in\mathbb{N}^*}$. E_A is of spectral dimension $d \in \mathbb{R}^{+,*}$ if $\sum_{i\in\mathbb{N}^*} b_i^2 < \infty$ and when $n \to \infty$:*

$$\sum_{i \geq n+1} b_i^2 = \mathcal{O}\big(n^{-\frac{2}{d}}\big).$$

The effective dimension of ellipsoid E_A is then $\sum_{i\in\mathbb{N}^*} b_i^2$.

The right notion of dimension for the control of metric entropy in infinite dimension is the spectral dimension, as shown in the next proposition: the ε-entropy of an ellipsoid scales as the spectral dimension in infinite dimension.

Proposition 10.3.1. *Let E_A be an ellipsoid in $\mathcal{V}$, of spectral dimension $d > 0$. We have, when $\varepsilon \to 0$:*

$$\mathcal{H}_\varepsilon(E_A) \leq d\ln\left(\varepsilon^{-1}\right)^2\big(1 + o(1)\big),$$

where $\mathcal{H}_\varepsilon(E_A)$ is the number (possibly infinite) of balls of radius ε required to cover E_A.

10.4. Statistical Preconditioning: Bounding Relative Condition Numbers and Uniform Concentration of Hessians

In this section, we present an application of Theorem 10.1 to optimization. Essentially, we show that statistical preconditioning-based optimization automatically benefits from low effective dimension in the data, thus proving a conjecture made in [HXB+20].

10.4.1. Large Deviation of Hessians

Let f be a convex function defined on $\mathbb{R}^d$. We assume that the following holds, which is true for logistic or ridge regressions.

Assumption 10.4.1 (Empirical Risk Structure)**.** *Let $\ell : \mathbb{R} \to \mathbb{R}$ convex, twice differentiable such that ℓ'' is $\|\ell''\|_{\text{Lip}}$-Lipschitz. Let $n \in \mathbb{N}^*$, some convex functions $\ell_j : \mathbb{R} \to \mathbb{R}, j \in [n]$ such that $\forall j \in [n], \ell_j'' = \ell''$ and* i.i.d. *Σ-subgaussian random variables $(a_j)_{j\in[n]}$. We assume that:*

$$\forall x \in \mathbb{R}^d, f(x) = \frac{1}{n}\sum_{j=1}^{n} \ell_j(a_j^\top x). \tag{10.18}$$

Proposition 10.4.1. *Denote H_x the Hessian of f at some point $x \in \mathbb{R}^d$ and $\bar{H}_x$ its mean. We have:*

$$H_x = \frac{1}{n}\sum_{i=1}^{n} \ell''(a_i^\top x) a_i a_i^\top, \text{ and } \bar{H}_x = \mathbb{E}_a\left[\ell''(a_1^\top x) a_1 a_1^\top\right].$$

Let:

$$Z = \sup_{\|x\|\leq 1} \left\|H_x - \bar{H}_x\right\|_{\text{op}}. \tag{10.19}$$

With probability $1-\delta$, we have, with C a universal constant:

$$Z \leq C\sigma_1^3 \left\|\ell''\right\|_{\text{Lip}} \left(\frac{(d_{\text{eff}}(3)\ln(d) + \ln(1/\delta))\sqrt{d_{\text{eff}}(1) + \ln(n/\delta)}}{n} + \frac{\sqrt{\ln(1/\delta)} + \sqrt{d_{\text{eff}}(1)\ln(d)}}{\sqrt{n}} \right).$$

Previous works [HXB+20] obtained:

$$C'\sigma_1^3 \left\|\ell''\right\|_{\text{Lip}} \frac{(d + \ln(1/\delta))\sqrt{d_{\text{eff}}(1) + \ln(n/\delta)}}{\sqrt{n}} \left[\frac{1}{\sqrt{d}} + \frac{1}{\sqrt{n}}\right]. \tag{10.20}$$

In order for this bound to be of order 1, n was required to be of order the whole dimension d, while we only need n to be of order $d_{\text{eff}}(3)$.

10.4.2. Statistical Preconditioning

Consider the following optimization problem:

$$\min_{x\in\mathbb{R}^d} \Phi(x) := F(x) + \psi(x), \tag{10.21}$$

where $F(x) = \frac{1}{n}\sum_{j=1}^{n} f_j(x)$ has a finite sum structure and ψ is a convex regularization function. Standard assumptions are the following:

$$\forall x, \sigma_F I_d \leq \nabla^2 F(x) \leq L_F I_d. \tag{10.22}$$

We focus on a basic setting of distributed optimization. At each iteration $t = 0, 1, ...,$ the server broadcasts the parameter x_t to all workers $j \in \{1, ..., n\}$. Each machine j then com-

putes in parallel $\nabla f_j(x_t)$ and sends it back to the server, who finally aggregates the gradients to form $\nabla F(x_t) = \frac{1}{n}\sum_j \nabla f_j(x_t)$ and use it to update x_t in the following way, using a standard proximal gradient descent, for some parameter $\eta_t \leq 1/L_F$:

$$x_{t+1} \in \arg\min_{x\in\mathbb{R}^d} \left\{ \langle \nabla F(x_t), x\rangle + \psi(x) + \frac{1}{2\eta_t}\|x - x_t\|^2 \right\}. \tag{10.23}$$

Setting $\eta_t = 1/L_F$ yields linear convergence:

$$\Phi(x_t) - \Phi(x^*) \leq L_F(1 - \kappa_F^{-1})^t \|x_0 - x^*\|^2. \tag{10.24}$$

In general, using an accelerated version of (10.23), one obtains a communication complexity (*i.e.* number of steps required to reach a precision $\varepsilon > 0$) of $O(\kappa_F^{1/2}\ln(1/\varepsilon))$ (where $\kappa_F = \frac{L_F}{\sigma_F}$) that cannot be improved in general. Statistical preconditioning is then a technique to improve each iteration's efficiency, based on the following insight: considering *i.i.d.* datasets leads to statistically similar local gradients ∇f_j. The essential tool for preconditioning is the Bregman divergence.

Definition 10.4.1 (Bregman divergence and Relative Smoothness)**.** *For a convex function $\phi : \mathbb{R}^d \to \mathbb{R}$, we define D_ϕ its Bregman divergence by:*

$$\forall x, y \in \mathbb{R}^d, D_\phi(x, y) = \phi(x) - \phi(y) - \langle \nabla\phi(y), x - y\rangle. \tag{10.25}$$

For convex functions $\phi, F : \mathbb{R}^d \to \mathbb{R}$, we say that F is relatively $L_{F/\phi}$-smooth and $\sigma_{F/\phi}$-strongly-convex if, for all $x, y \in \mathbb{R}^d$:

$$\sigma_{F/\phi} D_\phi(x, y) \leq D_F(x, y) \leq L_{F/\phi} D_\phi(x, y), \tag{10.26}$$

or equivalently:

$$\sigma_{F/\phi}\nabla^2\phi(x) \leq \nabla^2 F(x) \leq L_{F/\phi}\nabla^2\phi(x), \tag{10.27}$$

We consequently define $\kappa_{F/\phi} = \frac{L_{F/\phi}}{\sigma_{F/\phi}}$ the relative condition number of F with respect to ϕ.

Taking $\phi = \frac{1}{2}\|.\|^2$ gives $D_\phi = \frac{1}{2}\|.\|^2$ and thus yields classical smoothness and strong-convexity definitions. The idea of preconditioning is then to replace $\frac{1}{2\eta_t}\|x - x_t\|^2$ in (10.23) by $D_\phi(x, y)$ for a *convenient* function ϕ which the server has access to, leading to:

$$x_{t+1} \in \arg\min_{x\in\mathbb{R}^d} \left\{ \langle \nabla F(x_t), x\rangle + \psi(x) + \frac{1}{\eta_t} D_\phi(x, x_t) \right\}. \tag{10.28}$$

With $\eta_t = 1/L_{F/\phi}$, the sequence generated by (10.28) satisfies:

$$\Phi(x_t) - \Phi(x^*) \leq L_{F/\phi}(1 - \kappa_{F/\phi}^{-1})^t. \tag{10.29}$$

Hence, the effectiveness of preconditioning hinges on how smaller $\kappa_{F/\phi}$ is compared to κ_F. Next subsection presents how our large deviation bound of Hessians (Proposition 10.4.1) comes into place. The better ϕ approximates F, the smaller $\kappa_{F/\phi}$ and the more efficient each iteration of (10.28) is.

10.4.3. Main Results in Statistical Preconditioning

We furthermore assume that $F(x) = f(x) + \frac{\lambda}{2}\|x\|^2$ where f verifies Assumption 10.4.1 and $\lambda > 0$. Assume that the server has access to an *i.i.d.* sample $\tilde{a}_1, ..., \tilde{a}_N$ of the same law as the a_j's and to functions $\tilde{\ell}_1, ..., \tilde{\ell}_N$ such that $\tilde{\ell}_i'' = \ell''$. Define $\tilde{f}(x) = \frac{\lambda}{2}\|x\|^2 + \frac{1}{N}\sum_{i=1}^N \tilde{\ell}_i(a_i^\top x)$.

The preconditioner ϕ is chosen as, for some $\mu > 0$:

$$\phi(x) = \frac{\lambda}{2}\|x\|^2 + \frac{1}{N}\sum_{i=1}^{N} \tilde{\ell}_i(\tilde{a}_i^\top x) + \frac{\mu}{2}\|x\|^2, \tag{10.30}$$

Parameter $\mu > 0$ is chosen such that, with high probability:

$$\forall x \in \mathrm{Dom}_\psi, \left\|\nabla^2 \tilde{\mathrm{f}}(\mathrm{x}) - \nabla^2 \mathrm{F}(\mathrm{x})\right\|_{\mathrm{op}} \leq \mu. \tag{10.31}$$

For such a $\mu > 0$, we have: $L_{F/\phi} \leq 1$, $\sigma_{F/\phi} \geq (1 + 2\mu/\lambda)^{-1}$ and $\kappa_{F/\phi} \leq 1 + \frac{2\mu}{\lambda}$. Recall that for $t = 0, 1, 2, ...$, we have $\|x_t - x^*\|^2 \leq C(1 - \kappa_{F/\phi})^t$.

Proposition 10.4.2 (Statistical Preconditioning: Non-Isotropic Results)**.** *Assume that for all $x \in \mathrm{Dom}_\psi$, $\|x\| \leq R$. Under Assumption 10.4.1, with probability $1 - \delta$, we have:*

$$\sup_{\|x\| \leq R} \left\|\nabla^2 \tilde{f}(x) - \nabla^2 F(x)\right\| \leq CR\sigma_1^3 \left\|\ell''\right\|_{\mathrm{Lip}} \left(\frac{(d_{\mathrm{eff}}(3)\ln(d) + \ln(1/\delta))\sqrt{d_{\mathrm{eff}}(1) + \ln(n/\delta)}}{n} + \frac{\sqrt{\ln(1/\delta)} + \sqrt{d_{\mathrm{eff}}(1)\ln(d)}}{\sqrt{n}} \right).$$

If μ is taken as this upper bound, then we control the rate of convergence in (10.29) with:

$$\kappa_{F/\phi} = 1 + \tilde{O}\left\{ \frac{R\sigma_1^3 \|\ell''\|_{\mathrm{Lip}}}{\lambda} \max\left(\frac{\sqrt{d_{\mathrm{eff}}(1)}}{\sqrt{n}}, \frac{\sqrt{d_{\mathrm{eff}}(1)} d_{\mathrm{eff}}(3)}{n} \right) \right\}, \tag{10.32}$$

where $\tilde{O}$ hides logarithmic factors in d, n and δ^{-1}.

Contrast this with known results:

Remark 10.4.1 (Statistical Preconditioning: Isotropic Results)**.** *Still under Assumption 10.4.1, [HXB+20] obtained:*

$$\kappa_{F/\phi} = 1 + \tilde{O}\left\{ \frac{R\sigma_1^3 \|\ell''\|_{\mathrm{Lip}}}{\lambda} \max\left(\sqrt{\frac{d}{n}}, \frac{d^{3/2}}{n} \right) \right\}. \tag{10.33}$$

The only parameter required is an upper-bound on $d_{\mathrm{eff}}(3)$ and $d_{\mathrm{eff}}(1)$ in order to tune μ. Simply knowing that data are distributed according to a highly non-isotropic subgaussian law can thus improve the efficiency of statistical preconditioning, by decreasing drastically estimates of $\kappa_{F/\phi}$ and the number of samples required in the preconditioning function.

Chapter 11

(S)GD over diagonal linear networks and edge of stability

In this chapter, we investigate the impact of stochasticity and large stepsizes on the implicit regularisation of gradient descent (GD) and stochastic gradient descent (SGD) over 2-layer diagonal linear networks. We prove the convergence of GD and SGD with macroscopic stepsizes in an overparametrised regression setting and provide a characterisation of their solution through an implicit regularisation problem. Our characterisation provides insights on how the choice of minibatch sizes and stepsizes lead to qualitatively distinct behaviors in the solutions. Specifically, we show that for sparse regression learned with 2-layer diagonal linear networks, large stepsizes consistently benefit SGD, whereas they can hinder the recovery of sparse solutions for GD. These effects are amplified for stepsizes in a tight window just below the divergence threshold, known as the "edge of stability" regime.

This chapter is based on [EPGF23]; we refer the interested reader to the paper for the proofs that are not included in this manuscript. This chapter is to be put in perspective of Proposition 1.1.4 in the introduction chapter. The analysis and the results provided here are of the same flavor but for a non-convex model, and explicitly provide implicit regularization problems that are hyperparameters and trajectory dependent, highlighting a clear relation between generalization and optimization parameters hypertuning.

11.1. Introduction

The stochastic gradient descent algorithm (SGD) [RM51] is the foundational algorithm for almost all neural network training. Though a remarkably simple algorithm, it has led to many impressive empirical results and is a key driver of deep learning. However the performances of SGD are quite puzzling from a theoretical point of view as (1) its convergence is highly non-trivial and (2) there exist many global minimums for the training objective which generalise very poorly [ZBH+17].

To explain this second point, the concept of implicit regularisation has emerged: if overfitting is harmless in many real-world prediction tasks, it must be because the optimisation process is *implicitly favoring* solutions that have good generalisation properties for the task. The canonical example is overparametrised linear regression with more trainable parameters than number of samples: although there are infinitely many solutions that fit the samples, GD and SGD explore only a small subspace of all the possible parameters. As a result, it can be shown that they implicitly converge to the closest solution in terms of the ℓ_2 distance, and this without explicit regularisation [ZBH+17, GLSS18a].

Currently, most theoretical works on implicit regularisation have primarily focused on continuous time approximations of (S)GD where the impact of crucial hyperparameters such as the stepsize and the minibatch size are ignored. One such common simplification is to analyse gradient flow, which is a continuous time limit of GD and minibatch SGD with an infinitesimal stepsize. By definition, this analysis does not capture the effect of stepsize or stochasticity. Another approach is to approximate SGD by a stochastic gradient flow [Woj21, PPVF21], which tries to capture the noise and the stepsize using an appropriate

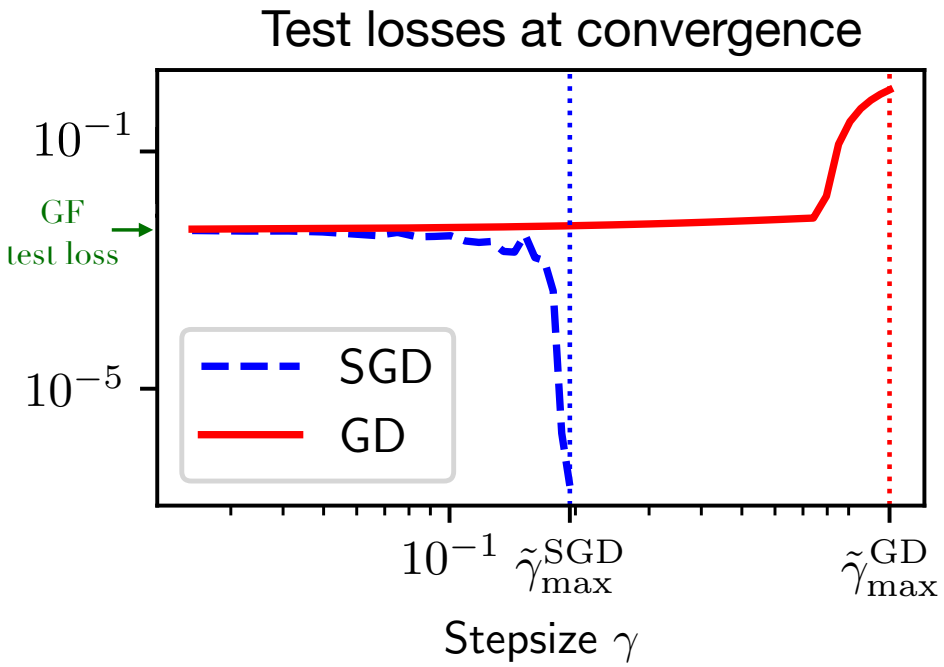


Figure 11.1 – *Noiseless sparse regression with a diagonal linear network using SGD and GD, with parameters initialized at the scale of* $\alpha = 0.1$ *(Section 11.2). The test losses at convergence for various stepsizes are plotted for GD and SGD. Small stepsizes correspond to gradient flow (GF) performance. We see that increasing the stepsize improves the generalisation properties of SGD, but deteriorates that of GD. The dashed vertical lines at stepsizes* $\tilde{\gamma}^{SGD}_{\max}$ *and* $\tilde{\gamma}^{GD}_{\max}$ *denote the largest stepsizes for which SGD and GD, respectively, converge. See Section 11.2 for the precise experimental setting.*

stochastic differential equation. However, there are no theoretical guarantees that these results can be transferred to minibatch SGD as used in practice. This is a limitation in our understanding since the performances of most deep learning models are often sensitive to the choice of stepsize and minibatch size. The importance of stepsize and SGD minibatch size is common knowledge in practice and has also been systematically established in controlled experiments [KMN+17a, ML18, GGP+22].

In this work, we aim to expand our understanding of the impact of stochasticity and stepsizes by analysing the (S)GD trajectory in 2-layer diagonal networks (DLNs). In Figure 11.1, we show that even in our simple network, there are significant differences between the nature of the solutions recovered by SGD and GD at macroscopic stepsizes. We discuss this behavior further in the later sections.

The 2-layer diagonal linear network which we consider is a simplified neural network that has received significant attention lately [WGL+20, VKR19, HWLM21, PVRF22]. Despite its simplicity, it surprisingly reveals training characteristics which are observed in much more complex architectures, such as the role of the initialisation [WGL+20], the role of noise [PPVF21, PVRF22], or the emergence of saddle-to-saddle dynamics [Ber22, PF23]. It therefore serves as an ideal proxy model for gaining a deeper understanding of complex phenomenons such as the roles of stepsizes and of stochasticity as highlighted in this chapter. We also point out that implicit bias and convergence for more complex architectures such as 2-layer ReLU networks, matrix multiplication are not yet fully understood, even for the simple gradient flow. Therefore studying the subtler effects of large stepsizes and stochasticity in these settings is currently out of reach.

11.1.1. Main results and chapter organisation

The overparametrised regression setting and diagonal linear networks are introduced in Section 11.2. We formulate our theoretical results (Theorems 11.1 and 11.2) in Section 11.3:

we prove that for **macroscopic stepsizes**, gradient descent and stochastic gradient descent over 2-layer diagonal linear networks converge to a zero-training loss solution $\beta^\star_\infty$. We further provide a refined characterization of $\beta^\star_\infty$ through a trajectory-dependent implicit regularisation problem, that captures the effects of hyperparameters of the algorithm, such as stepsizes and batchsizes, in useful and analysable ways. In Section 11.4 we then leverage this crisp characterisation to explain the influence of crucial parameters such as the stepsize and batch-size on the recovered solution. Importantly **our analysis shows a stark difference between the generalisation performances of GD and SGD for large stepsizes**, hence explaining the numerical results seen in Figure 11.1 for the sparse regression setting. Finally, in Section 11.5, we use our results to shed new light on the *Edge of Stability* (*EoS*) phenomenon [CKL+21].

11.1.2. Related works

Implicit bias. The concept of implicit bias from optimization algorithm in neural networks has been studied extensively in the past few years, starting with early works of [Tel13, NTS14, KMN+17a, SHN+18]. The theoretical results on implicit regularisation have been extended to multiplicative parametrisations [GWB+17, GLSS18b], linear networks [JT19], and homogeneous networks [LL19, JT20, COB19]. For regression loss on diagonal linear networks studied in this work, [WGL+20] demonstrate that the scale of the initialisation determines the type of solution obtained, with large initialisations yielding minimum ℓ_2 norm solutions—the neural tangent kernel regime [JGH18] and small initialisation resulting in minimum ℓ_1 norm solutions—the *rich regime* [COB19]. The analysis relies on the link between gradient descent and mirror descent established by [GHS20] and further explored by [VKR20, WR20]. These works focus on full batch gradient, and often in the inifitesimal stepsize limit (gradient flow), leading to general insights and results that do not take into account the effects of stochasticity and large stepsizes.

The effect of stochasticity in SGD on generalisation. The relationship between stochasticity in SGD and generalisation has been studied in various works [MHB16, HHS17, CS18, KLY18, WME18]. Empirically, models generated by SGD exhibit better generalisation performance than those generated by GD [KMN+17b, JKA+17, HLT19]. Explanations related to the flatness of the minima picked by SGD have been proposed [HS97]. Label noise has been shown to influence the implicit bias of SGD [HWLM21, BGVV20, DML21, PVRF22] by implicitly regularising the sharp minimisers. Recently, studying a *stochastic gradient flow* that models the noise of SGD in continuous time with Brownian diffusion, [PPVF21] characterised for diagonal linear networks the limit of their stochastic process as the solution of an implicit regularisation problem. However similar explicit characterisation of the implicit bias remains unclear for SGD with large stepsizes.

The effect of stepsizes in GD and SGD. Recent efforts to understand how the choice of stepsizes affects the learning process and the properties of the recovered solution suggest that larger stepsizes lead to the minimisation of some notion of flatness of the loss function [SL18, KMN+17b, NRSS22, JKA+18, WME18, MMS21], backed by empirical evidences or stability analyses. Larger stepsizes have also been proven to be beneficial for specific architectures or problems: two-layer network [LWM19], regression [WZBG21], kernel regression [BMR22] or matrix factorisation [WCZT22]. For large stepsizes, it has been observed that GD enters an *Edge of Stability (EoS)* regime [CKL+21], in which the iterates and the train loss oscillate before converging to a zero-training error solution; this phenomenon has then been studied on simple toy models [ABC+22, ZWW+23, CB22, DNL23] for GD. Recently, [AVPVF22] presented empirical evidence that large stepsizes can lead to loss stabilisation and towards simpler predictors.

11.2. Setup and preliminaries

Overparametrised linear regression. We consider a linear regression over inputs X that write as $X = (x_1, \dots, x_n) \in (\mathbb{R}^d)^n$ and outputs $y = (y_1, \dots, y_n) \in \mathbb{R}^n$. We consider *overparametrised* problems where input dimension d is (much) larger than the number of samples n. In this case, there exists infinitely many linear predictors $\beta^\star \in \mathbb{R}^d$ which perfectly fit the training set, *i.e.*, $y_i = \langle \beta^\star, x_i \rangle$ for all $1 \leq i \leq n$. We call such vectors *interpolating predictors* or *interpolators* and we denote by $\mathcal{S}$ the set of all interpolators $\mathcal{S} = \{\beta^\star \in \mathbb{R}^d \text{ s.t. } \langle \beta^\star, x_i \rangle = y_i, \forall i \in [n]\}$. Note that $\mathcal{S}$ is an affine space of dimension greater than $d - n$ and equal to $\beta^\star + \operatorname{span}(x_1, \dots, x_n)^\perp$ for any $\beta^\star \in \mathcal{S}$. We consider the following quadratic loss: $\mathcal{L}(\beta) = \frac{1}{2n} \sum_{i=1}^n (\langle \beta, x_i \rangle - y_i)^2$, for $\beta \in \mathbb{R}^d$.

2-layer linear diagonal network. We parametrise regression vectors β as functions β_w of trainable parameters $w \in \mathbb{R}^p$. Although the final prediction function $x \mapsto \langle \beta_w, x \rangle$ is linear in the input x, the choice of the parametrisation drastically changes the solution recovered by the optimisation algorithm [GLSS18b]. In the case of the linear parametrisation $\beta_w = w$ many first-order methods (SGD, GD, with or without momentum) converge towards the same solution and the choice of stepsize does not impact the recovered solution beyond convergence. In an effort to better understand the effects of stochasticity and large stepsize, we consider the next simple parametrisation, that of a 2-layer diagonal linear neural network given by:

$$\beta_w = u \odot v \text{ where } w = (u, v) \in \mathbb{R}^{2d} . \tag{11.1}$$

This parametrisation can be viewed as a simple neural network $x \mapsto \langle u, \sigma(\mathrm{diag(v)x}) \rangle$ where the output weights are represented by u, the inner weights is the diagonal matrix $\mathrm{diag(v)}$, and the activation σ is the identity function. In this spirit, we refer to the entries of $w = (u, v) \in \mathbb{R}^{2d}$ as the *weights* and to $\beta = u \odot v \in \mathbb{R}^d$ as the *prediction parameter*. Despite the simplicity of the parametrisation (11.1), the loss function F over parameters $w = (u, v) \in \mathbb{R}^{2d}$ is **non-convex** (and thus the corresponding optimization problem is challenging to analyse), and is given by:

$$F(w) = \mathcal{L}(u \odot v) = \frac{1}{2n} \sum_{i=1}^n (y_i - \langle u \odot v, x_i \rangle)^2 . \tag{11.2}$$

Mini-batch SGD. We minimise F using mini-batch SGD: let $w_0 = (u_0, v_0)$ and for $k \geq 0$,

$$w_{k+1} = w_k - \gamma_k \nabla F_{\mathcal{B}_k}(w_k) , \quad \text{where} \quad F_{\mathcal{B}_k}(w) = \frac{1}{2b} \sum_{i \in \mathcal{B}_k} (y_i - \langle u \odot v, x_i \rangle)^2 , \tag{11.3}$$

where γ_k are stepsizes, $\mathcal{B}_k \subset [n]$ are mini-batches of $b \in [n]$ distinct samples sampled uniformly and independently, and $\nabla F_{\mathcal{B}_k}(w_k)$ are minibatch gradients of partial loss over $\mathcal{B}_k$, $F_{\mathcal{B}_k}(w) = \mathcal{L}_{\mathcal{B}_k}(u \odot v)$ defined above. Classical SGD and full-batch GD are special cases with $b = 1$ and $b = n$, respectively. For $k \geq 0$, we consider the successive prediction parameters $\beta_k = u_k \odot v_k$ built from the weights $w_k = (u_k, v_k)$. We analyse SGD initialised at $u_0 = \sqrt{2}\alpha \in \mathbb{R}^d_{>0}$ and $v_0 = \mathbf{0} \in \mathbb{R}^d$, resulting in $\beta_0 = \mathbf{0} \in \mathbb{R}^d$ independently of the chosen weight initialisation α [1].

Experimental details. We consider the noiseless sparse regression setting where $(x_i)_{i \in [n]} \sim \mathcal{N}(0, I_d)$ and $y_i = \langle \beta^\star_{\ell_1}, x_i \rangle$ for some s-sparse vector $\beta^\star_{\ell_1}$. We perform (S)GD over the DLN

[1] In [EPGF23], we show that the (S)GD trajectory with this initialisation exactly matches that of another common parametrisation $\beta_w = w_+^2 - w_-^2$ with initialisation $w_{+,0} = w_{-,0} = \alpha$. The second layer of our diagonal linear network is set to 0 in order to obtain results that are easier to interpret. However, our proof techniques can be applied directly to a general initialisation, at the cost of additional notations in our Theorems.

with a uniform initialisation $\alpha = \alpha\mathbf{1} \in \mathbb{R}^d$ where $\alpha > 0$. Figure 11.1 and Figure 11.2 (left) correspond to the setup $(n, d, s, \alpha) = (20, 30, 3, 0.1)$, Figure 11.2 (right) to $(n, d, s, \alpha) = (50, 100, 4, 0.1)$ and Figure 11.3 to $(n, d, s, \alpha) = (50, 100, 2, 0.1)$.

Notations. Let $H = \nabla^2\mathcal{L} = \frac{1}{n}\sum_i x_i x_i^\top$ denote the Hessian of $\mathcal{L}$, and for a batch $\mathcal{B} \subset [n]$ let $H_\mathcal{B} = \nabla^2\mathcal{L}_\mathcal{B} = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} x_i x_i^\top$ denote the Hessian of the partial loss over the batch $\mathcal{B}$. Let L denote the "smoothness" such that $\forall\beta$, $\|H_\mathcal{B}\beta\|_2 \le L\|\beta\|_2$, $\|H_\mathcal{B}\beta\|_\infty \le L\|\beta\|_\infty$ for all batches $\mathcal{B} \subset [n]$ of size b. A real function (e.g, $\log, \exp$) applied to a vector must be understood as element-wise application, and for vectors $u, v \in \mathbb{R}^d$, $u^2 = (u_i^2)_{i\in[d]}$, $u \odot v = (u_i v_i)_{i\in[d]}$ and $u/v = (u_i/v_i)_{i\in[d]}$. We write $\mathbf{1}$, $\mathbf{0}$ for the constant vectors with coordinates 1 and 0 respectively. The Bregman divergence [Bre67] of a differentiable convex function $h : \mathbb{R}^d \to \mathbb{R}$ is defined as $D_h(\beta_1, \beta_2) = h(\beta_1) - (h(\beta_2) + \langle\nabla h(\beta_2), \beta_1 - \beta_2\rangle)$.

11.3. Implicit bias of SGD and GD

We start by recalling some known results on the implicit bias of gradient flow on diagonal linear networks before presenting our main theorems on characterising the (stochastic) gradient descent solutions (theorem 11.1) as well as proving the convergence of the iterates (theorem 11.2).

11.3.1. Warmup: gradient flow

We first review prior findings on gradient flow on diagonal linear neural networks. [WGL+20] show that the limit β^*_α of the *gradient flow* $\mathrm{d}w_t = -\nabla F(w_t)\mathrm{d}t$ initialised at $(u_0, v_0) = (\sqrt{2}\alpha, \mathbf{0})$ is the solution of the minimal interpolation problem:

$$\beta^*_\alpha = \arg\min_{\beta^\star\in\mathcal{S}} \psi_\alpha(\beta^\star)\,, \quad \text{where} \quad \psi_\alpha(\beta) = \frac{1}{2}\sum_{i=1}^d \left(\beta_i \operatorname{arcsinh}(\frac{\beta_i}{\alpha_i^2}) - \sqrt{\beta_i^2 + \alpha_i^4} + \alpha_i^2\right). \quad (11.4)$$

The convex potential ψ_α is the **hyperbolic entropy function** (or **hypentropy**) [GHS20]. Depending on the structure of the vector α, the generalisation properties of $\beta^\star_\alpha$ highly vary. We point out the two main characteristics of α that affect the behaviour of ψ_α and therefore also the solution $\beta^\star_\alpha$.

1. The **Scale** of α. For an initialisation vector α we call the ℓ_1-norm $\|\alpha\|_1$ the **scale** of the initialisation. It is an important quantity affecting the properties of the recovered solution $\beta^\star_\alpha$. To see this let us consider a uniform initialisation of the form $\alpha = \alpha\mathbb{1}$ for a scalar value $\alpha > 0$. In this case the potential ψ_α has the property of resembling the ℓ_1-norm as the scale α vanishes: $\psi_\alpha \sim \ln(1/\alpha)\|.\|_1$ as $\alpha \to 0$. Hence, a small initialisation results in a low ℓ_1-norm solution which is known to induce sparse recovery guarantees [CRT06]. This setting is often referred to as the "rich" regime [WGL+20]. In contrast, using a large initialisation scale leads to solutions with low ℓ_2-norm: $\psi_\alpha \sim \|.\|_2^2/(2\alpha^2)$ as $\alpha \to \infty$, a setting known as the "kernel" or "lazy" regime. Overall, to retrieve the minimum ℓ_1-norm solution, one should use a uniform initialisation with small scale α [WGL+20, Theorem 2].

2. The **Shape** of α. In addition to the scale of the initialisation α, a lesser studied aspect is its "shape", which is a term we use to refer to the relative distribution of $\{\alpha_i\}_i$ along the d coordinates [AMN+21]. It is a crucial property because having $\alpha \to \mathbf{0}$ **does not** necessarily lead to the potential ψ_α being close to the ℓ_1-norm. Indeed, we have that $\psi_\alpha(\beta) \overset{\alpha\to\mathbf{0}}{\sim} \sum_{i=1}^d \ln(\frac{1}{\alpha_i})|\beta_i|$, therefore if the vector $\ln(1/\alpha)$ has entries changing at different rates, then $\psi_\alpha(\beta)$ is a **weighted** ℓ_1-norm. In words, if the entries of α *do not go to zero "uniformly"*, then the resulting implicit bias minimizes a weighed ℓ_1-norm. This phenomenon can lead to solutions with vastly different sparsity structure than the minimum ℓ_1-norm interpolator.

11.3.2. Implicit bias of (stochastic) gradient descent

In theorem 11.1, we prove that for an initialisation $\sqrt{2}\alpha \in \mathbb{R}^d$ and for **arbitrary** stepsize sequences $(\gamma_k)_{k\geq 0}$ **if the iterates converge to an interpolator**, then this interpolator is the solution of a constrained minimisation problem which involves the hyperbolic entropy ψ_{α_∞} defined in (11.4), where $\alpha_\infty \in \mathbb{R}^d$ is an effective initialisation which depends on the trajectory and on the stepsize sequence. Later, **we prove the convergence of iterates for macroscopic step sizes** in theorem 11.2.

Theorem 11.1 (Implicit bias of (S)GD)**.** *Let $(u_k, v_k)_{k\geq 0}$ follow the mini-batch SGD recursion (11.3) initialised at $(u_0, v_0) = (\sqrt{2}\alpha, \mathbf{0})$ and with stepsizes $(\gamma_k)_{k\geq 0}$. Let $(\beta_k)_{k\geq 0} = (u_k \odot v_k)_{k\geq 0}$ and assume that they converge to some interpolator $\beta^\star_\infty \in \mathcal{S}$. Then, $\beta^\star_\infty$ satisfies:*

$$\beta^\star_\infty = \underset{\beta^\star \in \mathcal{S}}{\arg\min}\ D_{\psi_{\alpha_\infty}}(\beta^\star, \tilde{\beta}_0)\,, \tag{11.5}$$

where $D_{\psi_{\alpha_\infty}}$ is the Bregman divergence with hyperentropy potential ψ_{α_∞} of the **effective initialisation** *α_∞, and $\tilde{\beta}_0$ is a small* **perturbation term**. *The* **effective initialisation** *α_∞ is given by,*

$$\alpha^2_\infty = \alpha^2 \odot \exp\left(-\sum_{k=0}^{\infty} q\big(\gamma_k \nabla \mathcal{L}_{\mathcal{B}_k}(\beta_k)\big)\right)\,, \tag{11.6}$$

where $q(x) = -\frac{1}{2}\ln((1-x^2)^2)$ satisfies $q(x) \geq 0$ for $|x| \leq \sqrt{2}$, with the convention $q(1) = +\infty$.

The **perturbation term** *$\tilde{\beta}_0 \in \mathbb{R}^d$ is explicitly given by $\tilde{\beta}_0 = \frac{1}{2}\big(\alpha^2_+ - \alpha^2_-\big)$, where $q_\pm(x) = \mp 2x - \ln((1 \mp x)^2)$, and $\alpha^2_\pm = \alpha^2 \odot \exp\big(-\sum_{k=0}^{\infty} q_\pm(\gamma_k \nabla \mathcal{L}_{\mathcal{B}_k}(\beta_k))\big)$.*

Trajectory-dependent characterisation. The characterisation of $\beta^\star_\infty$ in Theorem 11.1 holds for any stepsize schedule such that the iterates converge and goes beyond the continuous-time frameworks previously studied [WGL+20, PPVF21]. The result even holds for adaptive stepsize schedules which keep the stepsize scalar such as AdaDelta [Zei12]. An important aspect of our result is that α_∞ and $\tilde{\beta}_0$ depend on the iterates' trajectory. Nevertheless, we argue that our formulation provides useful ingredients for understanding the implicit regularisation effects of (S)GD for this problem compared to trivial characterisations (such as *e.g.*, $\min_\beta \|\beta - \beta^\star_\infty\|$). Importantly, **the key parameters $\alpha_\infty, \tilde{\beta}_0$ depend on crucial parameters such as the stepsize and noise in a useful and analysable manner**: understanding how they affect α_∞ and $\tilde{\beta}_0$ coincides with understanding how they affect the recovered solution $\beta^\star_\infty$ and its generalisation properties. This is precisely the object of Sections 11.4 and 11.5 where we discuss the qualitative and quantitative insights from Theorem 11.1 in greater detail.

The perturbation $\tilde{\beta}_0$ can be ignored. We have that under reasonable assumptions on the stepsizes, that $|\tilde{\beta}_0| \leq \alpha^2$ and $\alpha_\infty \leq \alpha$ (component-wise). The magnitude of $\tilde{\beta}_0$ is therefore negligible in front of the magnitudes of $\beta^\star \in S$ and one can roughly ignore the term $\tilde{\beta}_0$. Hence, the implicit regularisation eq. (11.5) can be thought of as $\beta^\star_\infty \approx \arg\min_{\beta^\star \in S} D_{\psi_{\alpha_\infty}}(\beta^\star, 0) = \psi_{\alpha_\infty}(\beta^\star)$, and thus *the solution $\beta^\star_\infty$ minimises the same potential function that the solution of gradient flow (see Equation (11.4)), but with an effective initialisation α_∞*. Also note that for $\gamma_k \equiv \gamma \to 0$ we have $\alpha_\infty \to \alpha$ and $\tilde{\beta}_0 \to \mathbf{0}$, recovering the previously known result for gradient flow (11.4).

Deviation from gradient flow. The difference with gradient flow is directly associated with the quantity $\sum_k q(\gamma_k \nabla \mathcal{L}_{\mathcal{B}_k}(\beta_k))$. Also, as the (stochastic) gradients converge to 0 and $q(x) \overset{x\to 0}{\sim} x^2$, one should think of this sum as roughly being $\sum_k \nabla \mathcal{L}_{\mathcal{B}_k}(\beta_k)^2$: the larger this sum, the more the recovered solution differs from that of gradient flow. The full picture

of how large stepsizes and stochasticity impact the generalisation properties of $\beta^\star_\infty$ and the recovery of minimum ℓ_1-norm solution is nuanced as clearly seen in fig. 11.1.

11.3.3. Convergence of the iterates

Theorem 11.1 provides the implicit minimisation problem but says nothing about the convergence of the iterates. Here we show under very reasonable assumptions on the stepsizes that the iterates indeed converge towards a global optimum. Note that since the loss F is non-convex, such a convergence result is non-trivial and requires an involved analysis.

Theorem 11.2 (Convergence of the iterates)**.** *Let $(u_k, v_k)_{k\geq 0}$ follow the mini-batch SGD recursion (11.3) initialised at $u_0 = \sqrt{2}\alpha \in \mathbb{R}^d_{>0}$ and $v_0 = \mathbf{0}$, and let $(\beta_k)_{k\geq 0} = (u_k \odot v_k)_{k\geq 0}$. Recall the "smoothness" parameter L on the minibatch loss defined in the notations. There exist $B > 0$ verifying $B = \tilde{\mathcal{O}}(\min_{\beta^\star\in\mathcal{S}} \|\beta^\star\|_\infty)$ and a numerical constant $c > 0$ such that for stepsizes satisfying $\gamma_k \leq \frac{c}{LB}$, the iterates $(\beta_k)_{k\geq 0}$ converge almost surely to the interpolator $\beta^\star_\infty$ solution of Equation (11.5).*

In fact, we can be more precise by showing an exponential rate of convergence of the losses as well as characterise the rate of convergence of the iterates as follows.

Proposition 11.3.1 (Quantitative convergence rates)**.** *For a uniform initialisation $\alpha = \alpha\mathbf{1}$ and under the assumptions of Theorem 11.2, we have:*

$$\mathbb{E}\left[\mathcal{L}(\beta_k)\right] \leq \left(1 - \frac{1}{2}\gamma\alpha^2\lambda_b\right)^k \mathcal{L}(\beta_0) \quad \text{and} \quad \mathbb{E}\left[\left\|\beta_k - \beta^\star_{\alpha_k}\right\|^2\right] \leq C\left(1 - \frac{1}{2}\gamma\alpha^2\lambda_b\right)^k,$$

where $\lambda_b > 0$ is the largest value such that $\lambda_b H \preceq \mathbb{E}_{\mathcal{B}}[H_{\mathcal{B}}]$,

$$C = 2B(\alpha^2\lambda^+_{\min})^{-1}\left(1 + (4B\lambda_{\max})(\alpha^2\lambda^+_{\min})^{-1}\right)\mathcal{L}(\beta_0)\,,$$

and $\lambda^+_{\min}, \lambda_{\max} > 0$ are respectively the smallest non-null and the largest eigenevalues of H, and $\beta^\star_{\alpha_k}$ is the interpolator that minimises the perturbed hypentropy h_k of parameter α_k, as defined in Equation (11.7) in the next subsection.

The convergence of the losses is proved directly using the time-varying mirror structure that we exhibit in the next subsection, the convergence of the iterates is proved by studying the curvature of the mirror maps on a small neighborhood around the affine interpolation space.

11.3.4. Sketch of proof through a time varying mirror descent

As in the continuous-time framework, our results heavily rely on showing that the iterates $(\beta_k)_k$ follow a mirror descent recursion with time-varying potentials on the convex loss $\mathcal{L}(\beta)$. To show this, we first define the following quantities:

$$\alpha_k^2 = \alpha_{+,k} \odot \alpha_{-,k} \qquad \text{and} \qquad \phi_k = \frac{1}{2}\operatorname{arcsinh}\left(\frac{\alpha^2_{+,k} - \alpha^2_{-,k}}{2\alpha_k^2}\right) \in \mathbb{R}^d\,,$$

where $\alpha_{\pm,k} = \alpha \exp\left(-\frac{1}{2}\sum_{i=0}^{k-1} q_\pm\big(\gamma_\ell \nabla\mathcal{L}_{\mathcal{B}_\ell}(\beta_\ell)\big)\right) \in \mathbb{R}^d$. Finally for $k \geq 0$, we define the potentials $(h_k : \mathbb{R}^d \to \mathbb{R})_{k\geq 0}$ as:

$$h_k(\beta) = \psi_{\alpha_k}(\beta) - \langle \phi_k, \beta\rangle. \tag{11.7}$$

Where ψ_{α_k} is the hyperbolic entropy function defined Equation (11.4). Now that all the relevant quantities are defined, we can state the following proposition which explicits the time-varying stochastic mirror descent.

Proposition 11.3.2. *The iterates $(\beta_k = u_k \odot v_k)_{k\geq 0}$ from Equation (11.3) satisfy the Stochastic Mirror Descent recursion with varying potentials $(h_k)_k$:*

$$\nabla h_{k+1}(\beta_{k+1}) = \nabla h_k(\beta_k) - \gamma_k \nabla \mathcal{L}_{\mathcal{B}_k}(\beta_k)\,,$$

where $h_k : \mathbb{R}^d \to \mathbb{R}$ for $k \geq 0$ are defined Equation (11.7). Since $\nabla h_0(\beta_0) = 0$ we have:

$$\nabla h_k(\beta_k) \in \operatorname{span}(x_1, \ldots, x_n). \tag{11.8}$$

Theorems 11.1 and 11.2 and proposition 11.3.1 follow from this key proposition: by suitably modifying classical convex optimization techniques to account for the time-varying potentials, we can prove the convergence of the iterates towards an interpolator $\beta^\star_\infty$ along with that of the relevant quantities $\alpha_{\pm,k}$, α_k and ϕ_k. The implicit regularisation problem then directly follows from: (1) the limit condition $\nabla h_\infty(\beta_\infty) \in \operatorname{Span}(\mathrm{x}_1, \ldots, \mathrm{x}_n)$ as seen from eq. (11.8) and (2) the interpolation condition $X\beta^\star_\infty = y$. Indeed, these two conditions exactly correspond to the KKT conditions of the convex problem eq. (11.5).

11.4. Analysis of the impact of the stepsize and stochasticity on α_∞

In this section, we analyse the effects of large stepsizes and stochasticity on the implicit bias of (S)GD. We focus on how these factors influence the effective initialisation α_∞, which plays a key role as shown in Theorem 11.1. From its definition in eq. (11.6), we see that α_∞ is a function of the vector $\sum_k q(\gamma_k \nabla \mathcal{L}_{\mathcal{B}_k}(\beta_k))$. We henceforth call this quantity the *gain vector*. For simplicity of the discussions, from now on, we consider constant stepsizes $\gamma_k = \gamma$ for all $k \geq 0$ and a uniform initialisation of the weights $\alpha = \alpha \mathbf{1}$ with $\alpha > 0$. We can then write the gain vector as:

$$\mathrm{Gain}_\gamma = \ln\left(\frac{\alpha^2}{\alpha_\infty^2}\right) = \sum_{\mathrm{k}} \mathrm{q}(\gamma \nabla \mathcal{L}_{\mathcal{B}_\mathrm{k}}(\beta_\mathrm{k})) \in \mathbb{R}^\mathrm{d}\,.$$

Following our discussion in section 11.3.1 on the scale and the shape of α_∞, we recall the link between the scale and shape of Gain_γ and the recovered solution:

1. The **scale** of Gain_γ, i.e. the magnitude of $\|\mathrm{Gain}_\gamma\|_1$ indicates how much the implicit bias of (S)GD differs from that of gradient flow: $\|\mathrm{Gain}_\gamma\|_1 \sim 0$ implies that $\alpha_\infty \sim \alpha$ and therefore the recovered solution is close to that of gradient flow. On the contrary, $\|\mathrm{Gain}_\gamma\|_1 \gg \ln(1/\alpha)$ implies that α_∞ has effective scale much smaller than α thereby changing the implicit regularisation eq. (11.5).

2. The **shape** of Gain_γ indicates which coordinates of β in the associated minimum weighted ℓ_1 problem are most penalised. First recall from Section 11.3.1 that a uniformly large Gain_γ leads to ψ_{α_∞} being closer to the ℓ_1-norm. However, with small weight initialisation $\alpha \to 0$, we have,

$$\psi_{\alpha_\infty}(\beta) \sim \ln(\frac{1}{\alpha})\|\beta\|_1 + \sum_{i=1}^{d} \mathrm{Gain}_\gamma(\mathrm{i})|\beta_\mathrm{i}|\,, \tag{11.9}$$

   In this case, having a heterogeneously large vector Gain_γ leads to a weighted ℓ_1 norm as the effective implicit regularisation, where the coordinates of β corresponding to the largest entries of Gain_γ are less likely to be recovered.

11.4.1. The scale of Gain_γ is increasing with the stepsize

The following proposition highlights the dependencies of the scale of the gain $\|\mathrm{Gain}_\gamma\|_1$ in terms of various problem constants.

Proposition 11.4.1. *Let $\Lambda_b, \lambda_b > 0$ [2] be the largest and smallest values, respectively, such that $\lambda_b H \preceq \mathbb{E}_{\mathcal{B}}\left[H_{\mathcal{B}}^2\right] \preceq \Lambda_b H$. For any stepsize $\gamma > 0$ satisfying $\gamma \leq \frac{c}{BL}$ (as in theorem 11.2), initialisation $\alpha\mathbf{1}$ and batch size $b \in [n]$, the magnitude of the gain satisfies:*

$$\lambda_b \gamma^2 \sum_k \mathbb{E}\mathcal{L}(\beta_k) \leq \mathbb{E}\left[\|\text{Gain}_\gamma\|_1\right] \leq 2\Lambda_b \gamma^2 \sum_k \mathbb{E}\mathcal{L}(\beta_k)\,, \tag{11.10}$$

where the expectation is over a uniform and independent sampling of the batches $(\mathcal{B}_k)_{k\geq 0}$.

The slower the training, the larger the gain. eq. (11.10) shows that the slower the training loss converges to 0, the larger the sum of the loss and therefore the larger the scale of Gain_γ. This means that the (S)GD trajectory deviates from that of gradient flow if the stepsize and/or noise slows down the training. This supports observations previously made from stochastic gradient flow [PPVF21] analysis.

The bigger the stepsize, the larger the gain. The effect of the stepsize on the magnitude of the gain is not directly visible in eq. (11.10) because a larger stepsize tends to speed up the training. For stepsize $0 < \gamma \leq \gamma_{\max} = \frac{c}{BL}$ as in Theorem 11.2 we have that:

$$\sum_k \gamma^2 \mathcal{L}(\beta_k) = \Theta\left(\gamma \ln\left(\frac{1}{\alpha}\right) \left\|\beta^\star_{\ell_1}\right\|_1\right)\,. \tag{11.11}$$

eq. (11.11) clearly shows that increasing the stepsize **boosts** the magnitude $\|\text{Gain}_\gamma\|_1$ up until the limit of $\gamma_{\max}$. Therefore, the larger the stepsize the smaller is the effective scale of α_∞. In turn, larger gap between α_∞ and α leads to a larger deviation of (S)GD from the gradient flow.

Large stepsizes and Edge of Stability. The previous paragraph holds for stepsizes smaller than $\gamma_{\max}$ for which we can theoretically prove convergence. But what if we use even bigger stepsizes? Let $(\beta_k^\gamma)_k$ denote the iterates generated with stepsize γ and let us define $\tilde{\gamma}_{\max} = \sup_{\gamma\geq 0}\{\gamma \text{ s.t. } \forall \gamma' \in (0,\gamma),\ \sum_k \mathcal{L}(\beta_k^{\gamma'}) < \infty\}$, which corresponds to the largest stepsize such that the iterates still converge for a given problem (even if not provably so). From Proposition 11.4.1 we have that $\gamma_{\max} \leq \tilde{\gamma}_{\max}$. As we approach this upper bound on convergence $\gamma \to \tilde{\gamma}_{\max}$, the sum $\sum_k \mathcal{L}(\beta_k^\gamma)$ diverges. For such large stepsizes, the iterates of gradient descent tend to "bounce" and this regime is commonly referred to as the *Edge of Stability*. In this regime, the convergence of the loss can be made arbitrarily slow due to these bouncing effects. As a consequence, as seen through Equation (11.10), the magnitude of Gain_γ can be become arbitrarily big as observed in fig. 11.2 (left). In this regime, the recovered solution tends to dramatically differ from the gradient flow solution, as seen in fig. 11.1.

Impact of stochasticity and linear scaling rule. Assuming inputs x_i sampled from $\mathcal{N}(0, \sigma^2 I_d)$ with $\sigma^2 > 0$, we obtain $\mathbb{E}\left[\|\text{Gain}_\gamma\|_1\right] = \Theta\Big(\gamma \frac{\sigma^2 d}{b} \ln\left(\frac{1}{\alpha}\right) \|\beta^\star_{\ell_1}\|_1\Big)$, w.h.p. over the dataset. The scale of Gain_γ decreases with batch size and there exists a factor n between that of SGD and that of GD. Additionally, the magnitude of Gain_γ depends on $\frac{\gamma}{b}$, resembling the **linear scaling rule** commonly used in deep learning [GDG+17].

By analysing the magnitude $\|\text{Gain}_\gamma\|_1$, we have explained **the distinct behavior of (S)GD with large stepsizes compared to gradient flow**. However, our current analysis does not qualitatively distinguish the behavior between SGD and GD beyond the linear stepsize scaling

[2] $\Lambda_b, \lambda_b > 0$ are data-dependent constants; for $b = n$, we have $(\lambda_n, \Lambda_n) = (\lambda^+_{\min}(H), \lambda_{\max}(H))$ where $\lambda^+_{\min}(H)$ is the smallest non-null eigenvalue of H; for $b = 1$, we have $\min_i \|x_i\|_2^2 \leq \lambda_1 \leq \Lambda_1 \leq \max_i \|x_i\|_2^2$.

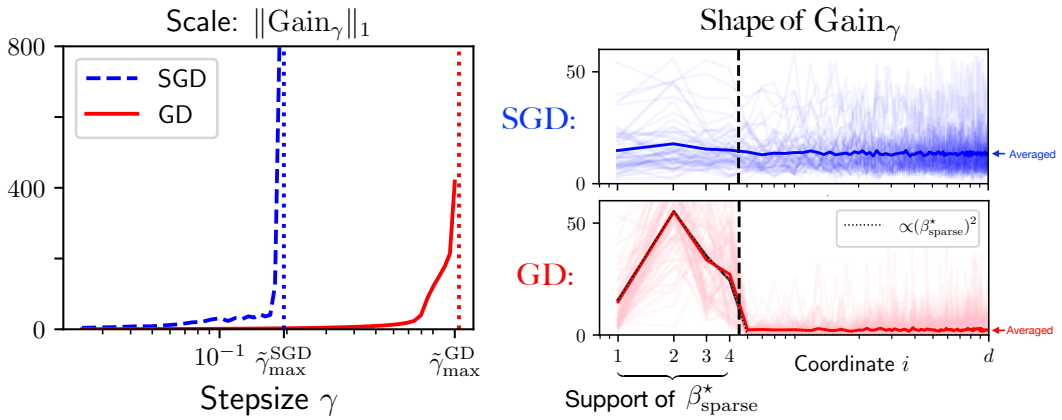


Figure 11.2 – *Left: the scale of* Gain_γ *explodes as* $\gamma \to \tilde{\gamma}_{\max}$ *for both GD and SGD. Right:* $\beta^\star_{\text{sparse}}$ *is fixed, we perform* 100 *runs of GD and SGD with different feature matrices, and we plot the* d *coordinates of* Gain_γ *(for GD and SGD) on the* x*-axis (which is in log scale for better visualisation). The shape of* $\text{Gain}^{\text{SGD}}_\gamma$ *is homogeneous whereas that of GD is heterogeneous with much higher magnitude on the support of* $\beta^\star_{\text{sparse}}$*. The shape of* $\text{Gain}^{\text{GD}}_\gamma$ *is proportional to the expected gradient at initialisation which is* $(\beta^\star_{\text{sparse}})^2$*.*

rules, in contrast with fig. 11.1. A deeper understanding of the shape of Gainγ is needed to explain this disparity.

11.4.2. The shape of Gain_γ explains the differences between GD and SGD

In this section, we restrict our presentation to single batch SGD ($b = 1$) and full batch GD ($b = n$). When visualising the typical shape of Gain_γ for large stepsizes (see Figure 11.2 - right), we note that GD and SGD behave very differently. For GD, the magnitude of Gain_γ is higher for coordinates in the support of $\beta^\star_{\ell_1}$ and thus these coordinates are adversely weighted in the asymptotic limit of ψ_{α_∞} (per (11.9)). This explains the distinction seed in fig. 11.1, where GD in this regime has poor sparse recovery despite having a small scale of α_∞, as opposed to SGD that behaves well.

The **shape** of Gain_γ is determined by the sum of the squared gradients $\sum_k \nabla\mathcal{L}_{\mathcal{B}_k}(\beta_k)^2$, and in particular by the degree of heterogeneity among the coordinates of this sum. Precisely analysing the sum over the whole trajectory of the iterates $(\beta_k)_k$ is technically out of reach. However, we empirically observe for the trajectories shown in Figure 11.2 that the shape is largely determined within the first few iterates as formalized in the observation below.

Observation 11.4.1. $\sum_k \nabla\mathcal{L}_{\mathcal{B}_k}(\beta_k)^2 \propto \mathbb{E}[\nabla\mathcal{L}_{\mathcal{B}_k}(\beta_0)^2]$.

In the simple case of a Gaussian noiseless sparse recovery problem (where $y_i = \langle\beta^\star_{\text{sparse}}, x_i\rangle$ for some sparse vector $\beta^\star_{\text{sparse}}$), we can control these gradients for GD and SGD as:

$$\nabla\mathcal{L}(\beta_0)^2 = (\beta^\star_{\text{sparse}})^2 + \varepsilon\,, \text{ for some } \varepsilon \text{ verifying } \|\varepsilon\|_\infty << \left\|\beta^\star_{\text{sparse}}\right\|^2_\infty\,, \tag{11.12}$$

$$\mathbb{E}_{i_0}[\nabla\mathcal{L}_{i_0}(\beta_0)^2] = \Theta\Big(\|\beta^\star_{\text{sparse}}\|_2^2\mathbf{1}\Big)\,. \tag{11.13}$$

The gradient of GD is heterogeneous. Since $\beta^\star_{\text{sparse}}$ is sparse by definition, we deduce from eq. (11.12) that $\nabla\mathcal{L}(\beta_0)$ is heterogeneous with larger values corresponding to the support of $\beta^\star_{\text{sparse}}$. This means that Gain_γ **has much larger values on the support of** $\beta^\star_{\text{sparse}}$. The corresponding weighted ℓ_1-norm therefore penalises the coordinates belonging to the support of $\beta^\star_{\text{sparse}}$, which hinders the recovery of $\beta^\star_{\text{sparse}}$.

The stochastic gradient of SGD is homogeneous. On the contrary, from eq. (11.13), we have that the initial stochastic gradients are homogeneous, leading to a weighted ℓ_1-norm where

the weights are roughly balanced. The corresponding weighted ℓ_1-norm is therefore close to the uniform ℓ_1-norm and the classical ℓ_1 recovery guarantees are expected.

Overall summary of the joint effects of the scale and shape. In summary we have the following trichotomy which fully explains Figure 11.1:

1. for small stepsizes, the scale is small, and (S)GD solutions are close to that of gradient flow;

2. for large stepsizes the scale is significant and the recovered solutions differ from GF:

   - for SGD the shape of α_∞ is uniform, the associated norm is closer to the ℓ_1-norm and the recovered solution is closer to the sparse solution;
   - for GD, the shape is heterogeneous, the associated norm is weighted such that it hinders the recovery of the sparse solution.

In this last section, we relate heuristically these findings to the *Edge of Stability* phenomenon.

11.5. Edge of Stability: the neural point of view

In recent years it has been noticed that when training neural networks with 'large' stepsizes at the limit of divergence, GD enters the *Edge of Stability (EoS)* regime. In this regime, as seen in Figure 11.3, the iterates of GD 'bounce' / 'oscillate'. In this section, we come back to the point of view of the weights $w_k = (u_k, v_k) \in \mathbb{R}^{2d}$ and make the connection between our previous results and the common understanding of the *EoS* phenomenon. The question we seek to answer is: in which case does GD enter the *EoS* regime, and if so, what are the consequences on the trajectory? *Keep in mind that this section aims to provide insights rather than formal statements.* We study the GD trajectory starting from a small initialisation $\alpha = \alpha\mathbf{1}$ where $\alpha \ll 1$ such that we can consider that gradient flow converges close to the sparse interpolator $\beta^\star_{\text{sparse}} = \beta_{w^\star_{\text{sparse}}}$ corresponding to the weights $w^\star_{\text{sparse}} = (\sqrt{|\beta^\star_{\text{sparse}}|}, \text{sign}(\beta^\star_{\text{sparse}})\sqrt{|\beta^\star_{\text{sparse}}|})$ (see Lemma 1 in [PF23] for the mapping from the predictors to weights for gradient flow). The trajectory of GD as seen in fig. 11.3 (left) can be decomposed into up to 3 phases.

First phase: gradient flow. The stepsize is appropriate for the local curvature (as seen in Figure 11.3, lower right) around initialisation and the iterates of GD remain close to the trajectory of gradient flow (in black in fig. 11.3). If the stepsize is such that $\gamma < \frac{2}{\lambda_{\max}(\nabla^2 F(w^\star_{\text{sparse}}))}$, then it is compatible with the local curvature and the iterates can converge: in this case GF and GD converge to the same point (as seen in fig. 11.1 for small stepsizes). For larger $\gamma > \frac{2}{\lambda_{\max}(\nabla^2 F(w^\star_{\text{sparse}}))}$ (as is the case for γ_{GD} in fig. 11.3, lower right), the iterates cannot converge to $\beta^\star_{\text{sparse}}$ and we enter the oscillating phase.

Second phase: oscillations. The iterates start oscillating. The gradient of F writes $\nabla_{(u,v)} F(w) \sim (\nabla\mathcal{L}(\beta) \odot v, \nabla\mathcal{L}(\beta) \odot u)$ and for w in the vicinity of $w^\star_{\text{sparse}}$ we have that $u_i \approx v_i \approx 0$ for $i \notin \text{supp}(\beta^\star_{\text{sparse}})$. Therefore for $w \sim w^\star_{\text{sparse}}$ we have that $\nabla_u F(w)_i \approx \nabla_v F(w)_i \approx 0$ for $i \notin \text{supp}(\beta^\star_{\text{sparse}})$ and the gradients roughly belong to $\text{Span}(\text{e}_\text{i}, \text{e}_{\text{i+d}})_{\text{i}\in\text{supp}(\beta^\star_{\text{sparse}})}$. This means that only the coordinates of the weights (u_i, v_i) for $i \in \text{supp}(\beta^\star_{\text{sparse}})$ can oscillate and similarly for $(\beta_i)_{i\in\text{supp}(\beta^\star_{\text{sparse}})}$ (as seen Figure 11.3 left).

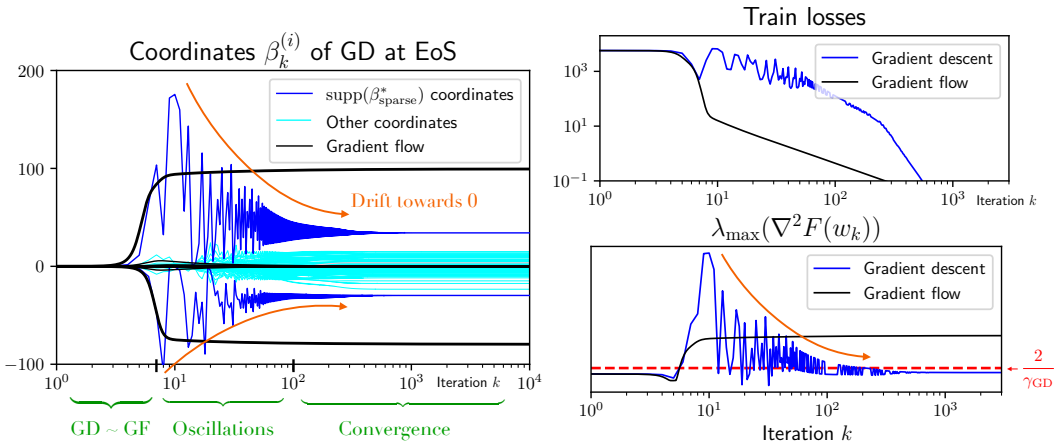


Figure 11.3 – *GD at the EoS. Left: For GD, the coordinates on the support of $\beta^\star_{\mathrm{sparse}}$ oscillate and drift towards 0. Right, top: The GD train losses saturate before eventually converging. Bottom: GF converges towards a solution that has a high hessian maximum eigenvalue. GD cannot converge towards this solution because of its large stepsize: it therefore drifts towards a solution that has a curvature just below $2/\gamma$.*

.

Last phase: convergence. Due to the oscillations, the iterates gradually drift towards a region of lower curvature (fig. 11.3, lower right, the sharpness decreases) where they may (potentially) converge. theorem 11.1 enables us to understand where they converge: the coordinates of β_k that have oscillated significantly along the trajectory belong to the support of $\beta^\star_{\mathrm{sparse}}$, and therefore $\mathrm{Gain}_\gamma(\mathrm{i})$ becomes much larger for $i \in \mathrm{supp}(\beta^\star_{\mathrm{sparse}})$ than for the other coordinates. Thus, the coordinates of the solution recovered in the *EoS* regime are heavily penalised on the support of the sparse solution. This is observed in Figure 11.3 (left): the oscillations of $(\beta_i)_{i\in\mathrm{supp}(\beta^\star_{\mathrm{sparse}})}$ lead to a gradual shift of these coordinates towards 0, hindering an accurate recovery of the solution $\beta^\star_{\mathrm{sparse}}$.

SGD in the EoS regime. In contrast to the behavior of GD where the oscillations primarily occur on the non-sparse coordinates of ground truth sparse model, for SGD we see a different behavior. For stepsizes in the *EoS* regime, just below the non-convergence threshold: the fluctuation of the coordinates occurs evenly over all coordinates, leading to a uniform α_∞. These fluctuations are reminiscent of label-noise SGD [AVPVF22], that have been shown to recover the sparse interpolator in diagonal linear networks [PVRF22].

Conclusion

This chapter studied the effect of stochasticity along with large stepsizes when training DLNs with (S)GD. We showed convergence of the iterates as well as explicitly characterise the recovered solution by exhibiting an implicit regularisation problem which depends on the iterates' trajectory. In essence the impact of stepsize and minibatch size are captured by the effective initialisation parameter α_∞ that depends on these choices in an informative way. We then used our characterisation to explain key empirical differences between SGD and GD and provide further insights on the role of stepsize and stochasticity. In particular, our characterisation explains the fundamentally different generalisation properties of SGD and GD solutions at large stepsizes as seen in Figure 11.1: without stochasticity, the use of large stepsizes can prevent the recovery of the sparse interpolator, even though the effective scale of the initialization decreases with larger stepsize for both SGD and GD. We also provide insights on the link between the *Edge of Stability* regime and our results.

We refer the interested reader to [EPGF23] for proofs of the results stated in this chapter and for more details.

Conclusion and perspectives

In this thesis, we have studied different flavours of distributed and collaborative learning, through the lenses of optimization theory for first-order methods, and statistical perspectives.

The first part of this thesis considered the acceleration of training algorithms: how to reach a global minima or approximate stationary points as fast as possible in wall-clock time. The first chapter of this part considered algorithmic improvements: making each iteration of gradient descent or of gossip algorithms as efficient as possible through mometum-based acceleration techniques. The three following parts directly considered wall-clock time acceleration and asynchronous techniques. Bridging these two perspectives — algorithmic acceleration and asynchronous speedups — remains open and is technically difficult to achieve, due to the non-robustness of accelerated algorithms to update changes. While we provided first quantitative asynchronous speedups for decentralized optimization, it remains to implement such algorithms in a large-scale setting in the training of large models. It is also unclear what communications are best in asynchronous environments: are our delayed gossip updates of Chapters 4 and 5 a good idea or is there better to do ?

The second part of this thesis then focused on privacy and personalization. On the privacy side, we introduced the first decentralized privacy-preserving mechanism: *Muffliato*. It serves as a building block for decentralized learning algorithms. There are many research directions to follow in this direction, the first direct ones being to improve the privacy analysis of Chapter 6, that we believe to be suboptimal due to its redundancy in the analysis and to the fact that our analysis is not taylored for optimization updates. Then, smarter privacy preserving decentralized mechanisms may exist, such as variations of *Muffliato* with correlated noise injections that anneal themselves on the long run, or clever communication randmoness analysis. On the personalization side, we provided optimal algorithms under similarity or structural assumptions. However, the assumptions we make in the two related chapters (Chapters 8 and 9) are convex optimization assumptions, therefore neglecting modern deep learning applications. Extending our analyses and ideas to the non-convex worlds remains an open and interesting question.

Finally, the last part of this thesis is not directly related to distributed optimization. It however puts the previous chapters in perspective of the implicit regularisation analysis we perform in Chapter 11: all previous chapters considered generalization or optimization separately, which may not always be the best approach in non-convex optimization. Bridging the gap between this chapter and the previous ones through generalization guarantees of our proposed distributed optimization algorithms would be an interesting question.

Bibliography

[AAF+20] Mahmoud Assran, Arda Aytekin, Hamid Reza Feyzmahdavian, Mikael Johansson, and Michael G. Rabbat. Advances in asynchronous parallel and distributed optimization. *Proceedings of the IEEE*, 108(11):2013–2031, 2020.

[ABC+22] Kwangjun Ahn, Sébastien Bubeck, Sinho Chewi, Yin Tat Lee, Felipe Suarez, and Yi Zhang. Learning threshold neurons via the "edge of stability". *arXiv preprint*, 2022.

[ABRW12] Alekh Agarwal, Peter L. Bartlett, Pradeep Ravikumar, and Martin J. Wainwright. Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization. *IEEE Transactions on Information Theory*, 58(5):3235–3249, 2012.

[ACD+22] Yossi Arjevani, Yair Carmon, John C. Duchi, Dylan J. Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, pages 1–50, 2022.

[ACG+16] Martín Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, October 2016. arXiv: 1607.00133.

[ACPR18] Hedy Attouch, Zaki Chbani, Juan Peypouquet, and Patrick Redont. Fast convergence of inertial dynamics and algorithms with asymptotic vanishing viscosity. *Mathematical Programming*, 168(1):123–175, 2018.

[ACR19] Hedy Attouch, Zaki Chbani, and Hassan Riahi. Rate of convergence of the Nesterov accelerated gradient method in the subcritical case $\alpha \leq 3$. *ESAIM: Control, Optimisation and Calculus of Variations*, 25:2, 2019.

[AD11] Alekh Agarwal and John C. Duchi. Distributed delayed stochastic optimization. *Advances in Neural Information Processing Systems*, 24, 2011.

[AEP06] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

[AF02] David Aldous and James Allen Fill. Reversible markov chains and random walks on graphs. 2002. Unfinished monograph, recompiled 2014, available at http://www.stat.berkeley.edu/$\sim$aldous/RWG/book.html.

[AGL+17] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in Neural Information Processing Systems*, 30, 2017.

[AHSL21] Rotem Zamir Aviv, Ido Hakimi, Assaf Schuster, and Kfir Yehuda Levy. Asynchronous distributed learning: Adapting to gradient delays without prior knowledge. In *Proceedings of the International Conference on Machine Learning*, volume 139, pages 436–445, 2021.

[AM06] Christophe Andrieu and Éric Moulines. On the ergodicity properties of some adaptive MCMC algorithms. *The Annals of Applied Probability*, 16(3), August 2006.

[AMN+21] Shahar Azulay, Edward Moroshko, Mor Shpigel Nacson, Blake E Woodworth, Nathan Srebro, Amir Globerson, and Daniel Soudry. On the implicit bias of initialization shape: Beyond infinitesimal mirror descent. In *International Conference on Machine Learning*, pages 468–477. PMLR, 2021.

[AMP05] Christophe Andrieu, Éric Moulines, and Pierre Priouret. Stability of stochastic approximation under verifiable conditions. *SIAM Journal on Control and Optimization*, 44(1):283–312, January 2005.

[ANW12] Alekh Agarwal, Sahand Negahban, and Martin J. Wainwright. Fast global convergence of gradient methods for high-dimensional statistical recovery. *The Annals of Statistics*, 40(5):2452 – 2482, 2012.

[AO17] Zeyuan Allen-Zhu and Lorenzo Orecchia. Linear Coupling: An Ultimate Unification of Gradient and Mirror Descent. In *Proceedings of the 8th Innovations in Theoretical Computer Science*, ITCS '17, 2017.

[AR21] Mahmoud S. Assran and Michael G. Rabbat. Asynchronous gradient push. *IEEE Transactions on Automatic Control*, 2021.

[AS19] Sulaiman A. Alghunaim and Ali H. Sayed. Linear convergence of primal-dual gradient methods and their performance in distributed optimization, 2019.

[ASS20] Yossi Arjevani, Ohad Shamir, and Nathan Srebro. A tight convergence analysis for stochastic gradient descent with delayed updates. In *Algorithmic Learning Theory*, pages 111–132. PMLR, 2020.

[ASSS16] Yossi Arjevani, Shai Shalev-Shwartz, and Ohad Shamir. On lower and upper bounds in smooth and strongly convex optimization. *Journal of Machine Learning Research*, 17(126):1–51, 2016.

[AVPVF22] M. Andriushchenko, A. Varre, L. Pillaud-Vivien, and N. Flammarion. SGD with large step sizes learns sparse features. *arXiv preprint*, 2022.

[AWBR09] Alekh Agarwal, Martin J. Wainwright, Peter L. Bartlett, and Pradeep K. Ravikumar. Information-theoretic lower bounds on the oracle complexity of convex optimization. In *Advances in Neural Information Processing Systems*, pages 1–9, 2009.

[Bau78] Gerard M. Baudet. Asynchronous iterative methods for multiprocessors. *Journal of the ACM (JACM)*, 25(2):226–244, 1978.

[BB07] Leon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, page 161–168, Red Hook, NY, USA, 2007. Curran Associates Inc.

[BBG18] Borja Balle, Gilles Barthe, and Marco Gaboardi. Privacy Amplification by Subsampling: Tight Analyses via Couplings and Divergences. In *NeurIPS*, 2018.

[BBG+20a] James Henry Bell, Kallista A. Bonawitz, Adrià Gascón, Tancrède Lepoint, and Mariana Raykova. Secure single-server aggregation with (poly)logarithmic overheads. In *CCS*, 2020.

[BBG20b] Raphaël Berthier, Francis Bach, and Pierre Gaillard. Accelerated gossip in networks of given dimension using jacobi polynomial iterations. *SIAM Journal on Mathematics of Data Science*, 2(1):24–47, 2020.

[BBG20c] Raphaël Berthier, Francis Bach, and Pierre Gaillard. Tight nonparametric convergence rates for stochastic gradient descent under the noiseless linear model. In *Advances in Neural Information Processing Systems*, volume 33, pages 2576–2586, 2020.

[BBGN19] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. Differentially Private Summation with Multi-Message Shuffling. Technical report, arxiv:1906.09116, 2019.

[BBL04] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. *Introduction to Statistical Learning Theory*, page 169–207. Springer Berlin Heidelberg, 2004.

[BBT17] Heinz H. Bauschke, Jérôme Bolte, and Marc Teboulle. A descent lemma beyond lipschitz gradient continuity: First-order methods revisited and applications. *Mathematics of Operations Research*, 42(2):330–348, 2017.

[BCE+23] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023.

[BCN18] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[BCV13] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, aug 2013.

[BD09] Thomas Blumensath and Mike E. Davies. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27(3):265–274, 2009.

[BE02] Olivier Bousquet and André Elisseeff. Stability and generalization. *J. Mach. Learn. Res.*, 2:499–526, mar 2002.

[Ber22] Raphaël Berthier. Incremental learning in diagonal linear networks. *arXiv preprint arXiv:2208.14673*, 2022.

[BGH20] Aurélien Bellet, Rachid Guerraoui, and Hadrien Hendrikx. Who started this rumor? Quantifying the natural differential privacy guarantees of gossip protocols. In *DISC*, 2020.

[BGHJ21] Martin Beaussart, Felix Grimberg, Mary-Anne Hartley, and Martin Jaggi. WAFFLE: Weighted Averaging for Personalized Federated Learning. *arXiv:2110.06978 [cs]*, October 2021.

[BGPS06] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE transactions on information theory*, 2006.

[BGTT18] Aurélien Bellet, Rachid Guerraoui, Mahsa Taziki, and Marc Tommasi. Personalized and private peer-to-peer machine learning. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 473–481, 2018.

[BGVV20] G. Blanc, N. Gupta, G. Valiant, and P. Valiant. Implicit regularization for deep neural networks driven by an Ornstein-Uhlenbeck like process. In *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pages 483–513. PMLR, 09–12 Jul 2020.

[BIK+17] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In *CCS*, 2017.

[BJW18] Michael Betancourt, Michael Jordan, and Ashia Wilson. On symplectic optimization. *arXiv preprint arXiv:1802.03653*, 2018.

[BKF22] Etienne Boursier, Mikhail Konobeev, and Nicolas Flammarion. Trace norm regularization for multi-task learning with scarce data. In Po-Ling Loh and Maxim Raginsky, editors, *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pages 1303–1327. PMLR, 02–05 Jul 2022.

[BKT97] N. G. Bean, F. P. Kelly, and P. G. Taylor. Braess's paradox in a loss network. *Journal of Applied Probability*, 34(1):155–159, 1997.

[BL23] Matthieu Blanke and Marc Lelarge. Flex: an adaptive exploration algorithm for nonlinear systems. In *International Conference on Machine Learning*, 2023.

[BLM13] Stéphane Boucheron, Gabor Lugosi, and Pascal Massart. *Concentration inequalities : a non asymptotic theory of independence*. Oxford University Press, 2013.

[BLN20] Sébastien Bubeck, Yuanzhi Li, and Dheeraj Nagaraj. A law of robustness for two-layers neural networks. *arXiv preprint arXiv:2009.14444*, 2020.

[BLS15] Sébastien Bubeck, Yin Tat Lee, and Mohit Singh. A geometric alternative to Nesterov's accelerated gradient descent. *arXiv preprint arXiv:1506.08187*, 2015.

[BM03] Samuel Burer and Renato D.C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, February 2003.

[BM04] Samuel Burer and Renato D.C. Monteiro. Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming*, 103(3):427–444, December 2004.

[BMP90] Albert Benveniste, Michel Métivier, and Pierre Priouret. *Adaptive Algorithms and Stochastic Approximations*. Springer Berlin Heidelberg, 1990.

[BMR22] G. Beugnot, J. Mairal, and A. Rudi. On the benefits of large learning rates for kernel methods. In *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pages 254–282. PMLR, 02–05 Jul 2022.

[BNH19] Tal Ben-Nun and Torsten Hoefler. Demystifying parallel and distributed deep learning: An in-depth concurrency analysis. *ACM Computing Surveys (CSUR)*, 52(4):1–43, 2019.

[Bor76] Libatius Borage. *Advanced Potion Making.* 1976.

[BR85] S.D. Brown and S.C. Rutan. Adaptive kalman filtering. *Journal of Research of the National Bureau of Standards*, 90(6):403, November 1985.

[Bre67] L.M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217, 1967.

[BRW+23] Marco Bornstein, Tahseen Rabbani, Evan Z Wang, Amrit Bedi, and Furong Huang. SWIFT: Rapid decentralized federated learning via wait-free model communication. In *ICLR*, 2023.

[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*, pages 464–473, Philadelphia, PA, USA, October 2014. IEEE.

[Bub15] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Found. Trends Mach. Learn.*, 8(3–4):231–357, November 2015.

[Car97] Rich Caruana. Multi-task learning. *Machine Learning*, 28(1):41–75, 1997.

[CB20] Edwige Cyffers and Aurélien Bellet. Privacy amplification by decentralization, 2020.

[CB22] Lei Chen and Joan Bruna. On gradient descent convergence beyond the edge of stability, 2022.

[CBLPP22] Nicolò Cesa-Bianchi, Pierre Laforgue, Andrea Paudice, and Massimiliano Pontil. Multitask online mirror descent. *Transactions on Machine Learning Research*, 2022.

[Cc22] Aakanksha Chowdhery and coauthors. Palm: Scaling language modeling with pathways, 2022.

[CCA+21] Zheng Chai, Yujing Chen, Ali Anwar, Liang Zhao, Yue Cheng, and Huzefa Rangwala. FedAT: a high-performance and communication-efficient federated learning system with asynchronous tiers. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16, 2021.

[CCD21] Gary Cheng, Karan Chadha, and John Duchi. Fine-tuning is Fine in Federated Learning. *arXiv:2108.07313 [cs, math, stat]*, August 2021.

[CDD+21] Alon Cohen, Amit Daniely, Yoel Drori, Tomer Koren, and Mariano Schain. Asynchronous stochastic optimization robust to arbitrary delays. *Advances in Neural Information Processing Systems*, 34, 2021.

[CDHS17] Yair Carmon, John C. Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points I. *arXiv preprint arXiv:1710.11606*, 2017.

[CDHS21] Yair Carmon, John C. Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points ii: First-order methods. *Math. Program.*, 185(1–2):315–355, jan 2021.

[CEBM22] Edwige Cyffers, Mathieu Even, Aurélien Bellet, and Laurent Massoulié. Muffliato: Peer-to-peer privacy amplification for decentralized optimization and averaging. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[CHMS21] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 2089–2099. PMLR, 18–24 Jul 2021.

[CHMS22] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Fedavg with fine tuning: Local updates lead to representation learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[CHS+23] Liam Collins, Hamed Hassani, Mahdi Soltanolkotabi, Aryan Mokhtari, and Sanjay Shakkottai. Provable multi-task representation learning by two-layer relu neural networks, 2023.

[CKL+21] Jeremy Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. In *International Conference on Learning Representations*, 2021.

[CKS+21] El Mahdi Chayti, Sai Praneeth Karimireddy, Sebastian U. Stich, Nicolas Flammarion, and Martin Jaggi. Linear speedup in personalized collaborative learning, 2021.

[CLD+18] Fei Chen, Mi Luo, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated meta-learning with fast convergence and efficient communication, 2018.

[CMOS22] Liam Collins, Aryan Mokhtari, Sewoong Oh, and Sanjay Shakkottai. MAML and ANIL provably learn representations. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 4238–4310. PMLR, 17–23 Jul 2022.

[COB19] Lénaïc Chizat, Edouard Oyallon, and Francis Bach. *On Lazy Training in Differentiable Programming.* 2019.

[CP11] Emmanuel J. Candès and Yaniv Plan. Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements. *IEEE Transactions on Information Theory*, 57(4):2342–2359, 2011.

[CPM+16] Jianmin Chen, Xinghao Pan, Rajat Monga, Samy Bengio, and Rafal Jozefowicz. Revisiting distributed synchronous SGD. *arXiv preprint arXiv:1604.00981*, 2016.

[CRT06] E. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006.

[CS18] Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In *International Conference on Learning Representations*, 2018.

[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.

[CSSS11] Andrew Cotter, Ohad Shamir, Nati Srebro, and Karthik Sridharan. Better mini-batch algorithms via accelerated gradient methods. In *Advances in Neural Information Processing Systems 24*, pages 1647–1655, 2011.

[CSU+19] Albert Cheu, Adam D. Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed Differential Privacy via Shuffling. In *EUROCRYPT*, 2019.

[CSY06] Ming Cao, Daniel A. Spielman, and Edmund M. Yeh. Accelerated gossip algorithms for distributed computation. In *44th Annual Allerton Conference on Communication, Control, and Computation*, pages 952–959, 2006.

[CT05] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley, April 2005.

[CYH+19] Hsin-Pai Cheng, Patrick Yu, Haojing Hu, Syed Zawad, Feng Yan, Shiyu Li, Hai Helen Li, and Yiran Chen. Towards Decentralized Deep Learning with Differential Privacy. In *CLOUD*, 2019.

[DAJJ11] John C. Duchi, Alekh Agarwal, Mikael Johansson, and Michael I. Jordan. Ergodic mirror descent. In *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 701–706, 2011.

[Dav84] Mark HA Davis. Piecewise-deterministic markov processes: a general class of non-diffusion stochastic models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 46(3):353–376, 1984.

[Dav18] Mark HA Davis. *Markov models & optimization*. Routledge, 2018.

[DBLJ14] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[DCGP19] Giulia Denevi, Carlo Ciliberto, Riccardo Grazzi, and Massimiliano Pontil. Learning-to-learn stochastic gradient descent with biased regularization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1566–1575. PMLR, 09–15 Jun 2019.

[DCM+12] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. *Advances in Neural Information Processing Systems*, 25, 2012.

[DEH21] Radu Alexandru Dragomir, Mathieu Even, and Hadrien Hendrikx. Fast stochastic bregman gradient methods: Sharp analysis and variance reduction. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 2815–2825. PMLR, 18–24 Jul 2021.

[DFHT22] John Duchi, Vitaly Feldman, Lunjia Hu, and Kunal Talwar. Subspace recovery from heterogeneous data with non-isotropic noise. *Advances in Neural Information Processing Systems*, 2022.

[DGBSX12] Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13(Jan):165–202, 2012.

[DGJ17] David L. Donoho, Matan Gavish, and Iain M. Johnstone. Optimal shrinkage of eigenvalues in the spiked covariance model. *arXiv preprint arXiv:1311.0851*, 2017.

[DHK+21] Simon Shaolei Du, Wei Hu, Sham M. Kakade, Jason D. Lee, and Qi Lei. Few-shot learning via learning the representation, provably. In *International Conference on Learning Representations*, 2021.

[DJW13] John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy and statistical minimax rates. In *FOCS*, 2013.

[DK21a] Kate Donahue and Jon Kleinberg. Model-sharing games: Analyzing federated learning under voluntary participation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6):5303–5311, May 2021.

[DK21b] Kate Donahue and Jon Kleinberg. Optimality and stability in federated learning: A game-theoretic approach. In *Advances in Neural Information Processing Systems*, 2021.

[DKM+06] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *EUROCRYPT*, 2006.

[DKM+10] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione. Gossip algorithms for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864, 2010.

[DKM20] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive Personalized Federated Learning. *arXiv:2003.13461 [cs, stat]*, November 2020. arXiv: 2003.13461.

[DKMM23] Yuyang Deng, Mohammad Mahdi Kamani, Pouria Mahdavinia, and Mehrdad Mahdavi. Distributed personalized empirical risk minimization, 2023.

[DL22] Ron Dorfman and Kfir Yehuda Levy. Adapting to mixing time in stochastic optimization with Markovian data. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5429–5446. PMLR, 17–23 Jul 2022.

[DML21] Alex Damian, Tengyu Ma, and Jason D. Lee. Label noise SGD provably prefers flat global minimizers. In *Advances in Neural Information Processing Systems*, 2021.

[DNL23] Alex Damian, Eshaan Nichani, and Jason D. Lee. Self-stabilization: The implicit bias of gradient descent at the edge of stability. In *International Conference on Learning Representations*, 2023.

[DO19] Jelena Diakonikolas and Lorenzo Orecchia. The approximate duality gap technique: A unified theory of first-order methods. *SIAM Journal on Optimization*, 29(1):660–689, 2019.

[Doa23] Thinh T. Doan. Finite-time analysis of markov gradient descent. *IEEE Transactions on Automatic Control*, 68(4):2140–2153, 2023.

[Doo90] J. L. Doob. *Stochastic Processes*. John Wiley & Sons, 1990.

[DPP04] I. Dumer, M. Pinsker, and V. V. Prelov. On coverings of ellipsoids in euclidean spaces. *IEEE Transactions on Information Theory*, 50:2348–2356, 2004.

[DR14] Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[DR19] John Duchi and Ryan Rogers. Lower bounds for locally private estimation via communication complexity. In Alina Beygelzimer and Daniel Hsu, editors, *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 1161–1191, Phoenix, USA, June 2019. PMLR.

[DSSSC08] John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the l1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 272–279, New York, NY, USA, 2008. Association for Computing Machinery.

[dST21] Alexandre d'Aspremont, Damien Scieur, and Adrien Taylor. Acceleration methods. 2021.

[DSW08] Alexandros DG Dimakis, Anand D Sarwate, and Martin J Wainwright. Geographic gossip: Efficient averaging for sensor networks. *IEEE Transactions on Signal Processing*, 56(3):1205–1216, 2008.

[Dud67] R.M Dudley. The sizes of compact subsets of hilbert space and continuity of gaussian processes. *Journal of Functional Analysis*, 1(3):290–330, 1967.

[Dud74] R.M Dudley. Metric entropy of some classes of sets with differentiable boundaries. *Journal of Approximation Theory*, 10(3):227–236, 1974.

[DW13] John C. Duchi and Martin J. Wainwright. Distance-based and continuum fano inequalities with applications to statistical estimation. 2013.

[DW22] Shu Ding and Wei Wang. Collaborative learning by detecting collaboration partners. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[EBB+21] Mathieu Even, Raphaël Berthier, Francis Bach, Nicolas Flammarion, Hadrien Hendrikx, Pierre Gaillard, Laurent Massoulié, and Adrien Taylor. Continuized accelerations of deterministic and stochastic gradient descents, and of gossip algorithms. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 28054–28066. Curran Associates, Inc., 2021.

[EFM+19] Ulfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, and Kunal Talwar. Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity. In *SODA*, 2019.

[EHM20] Mathieu Even, Hadrien Hendrikx, and Laurent Massoulié. Asynchrony and acceleration in gossip algorithms. *arXiv preprint arXiv:2011.02379*, 2020.

[EHM21a] Mathieu Even, Hadrien Hendrikx, and Laurent Massoulié. Asynchronous speedup in decentralized optimization. *arXiv preprint arXiv:2106.03585*, 2021.

[EHM21b] Mathieu Even, Hadrien Hendrikx, and Laurent Massoulié. Asynchrony and Acceleration in Gossip Algorithms. *arXiv:2011.02379 [cs, math]*, February 2021. arXiv: 2011.02379.

[EK10] David Easley and Jon Kleinberg. *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*. Cambridge University Press, 2010.

[EKM23] Mathieu Even, Anastasia Koloskova, and Laurent Massoulié. Asynchronous sgd on graphs: a unified framework for asynchronous decentralized and federated optimization. 2023.

[EM21] Mathieu Even and Laurent Massoulie. Concentration of non-isotropic random tensors with applications to learning and empirical risk minimization. In Mikhail Belkin and Samory Kpotufe, editors, *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134 of *Proceedings of Machine Learning Research*, pages 1847–1886. PMLR, 15–19 Aug 2021.

[EMS22a] Mathieu Even, Laurent Massoulié, and Kevin Scaman. On sample optimality in personalized collaborative and federated learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[EMS22b] Mathieu Even, Laurent Massoulié, and Kevin Scaman. Sample optimality and all-for-all strategies in personalized federated and collaborative learning. Technical report, arXiv:2201.13097, 2022.

[EPGF23] Mathieu Even, Scott Pesme, Suriya Gunasekar, and Nicolas Flammarion. (s)gd over diagonal linear networks: Implicit bias, large stepsizes and edge of stability. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 29406–29448. Curran Associates, Inc., 2023.

[ER59] P. Erdös and A. Rényi. On random graphs i. *Publicationes Mathematicae Debrecen*, 6:290, 1959.

[ESV+11] W. Ellens, F.M. Spieksma, P. Van Mieghem, A. Jamakovic, and R.E. Kooij. Effective graph resistance. *Linear Algebra and its Applications*, 435(10):2491–2506, 2011. Special Issue in Honor of Dragos Cvetkovic.

[Eve23] Mathieu Even. Stochastic gradient descent under markovian sampling schemes. In *ICML*, ICML'23, 2023.

[FB15] Nicolas Flammarion and Francis Bach. From averaging to acceleration, there is only a step-size. In *Conference on Learning Theory*, pages 658–695. PMLR, 2015.

[FMO20] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach. *Advances in Neural Information Processing Systems*, 33:3557–3568, 2020.

[FMSV16] G. Fort, E. Moulines, A. Schreck, and M. Vihola. Convergence of markovian stochastic approximation with discontinuous dynamics. *SIAM Journal on Control and Optimization*, 54(2):866–893, January 2016.

[FMT20] Vitaly Feldman, Audra McMillan, and Kunal Talwar. Hiding Among the Clones: A Simple and Nearly Optimal Analysis of Privacy Amplification by Shuffling. Technical report, arXiv:2012.12803, 2020.

[FMTT18a] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. Privacy Amplification by Iteration. In *FOCS*, 2018.

[FMTT18b] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. Privacy amplification by iteration. *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, Oct 2018.

[FS18] Simon Foucart and Srinivas Subramanian. Iterative hard thresholding for low-rank recovery from rank-one projections, 2018.

[FZ21] Vitaly Feldman and Tijana Zrnic. Individual Privacy Accounting via a Rényi Filter. In *NeurIPS*, 2021.

[GCYR20] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[GDG+17] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[GG23] Guillaume Garrigos and Robert M. Gower. Handbook of convergence theorems for (stochastic) gradient methods, 2023.

[GGK+20] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. Pure Differentially Private Summation from Anonymous Messages. Technical report, arXiv:2002.01919, 2020.

[GGP+22] Jonas Geiping, Micah Goldblum, Phillip E Pope, Michael Moeller, and Tom Goldstein. Stochastic training is not necessary for generalization. In *International Conference on Learning Representations*, 2022.

[GHKJ21] Felix Grimberg, Mary-Anne Hartley, Sai P. Karimireddy, and Martin Jaggi. Optimal Model Averaging: Towards Personalized Collaborative Learning. *arXiv:2110.12946 [cs, stat]*, October 2021. arXiv: 2110.12946.

[GHS20] Udaya Ghai, Elad Hazan, and Yoram Singer. Exponentiated gradient meets gradient descent. In *Proceedings of the 31st International Conference on Algorithmic Learning Theory*, volume 117 of *Proceedings of Machine Learning Research*, pages 386–407. PMLR, 08 Feb–11 Feb 2020.

[Gir21] Christophe Giraud. *Introduction to High-Dimensional Statistics*. Chapman and Hall/CRC, August 2021.

[GL09] Keqin Gu and Yi Liu. Lyapunov–krasovskii functional for uniform stability of coupled differential-functional equations. *Automatica*, 2009.

[GL13] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[GLQ+19] Robert M. Gower, Nicolas Loizou, Xun Qian, Alibek Sailanbayev, Egor Shulgin, and Peter Richtárik. SGD: General analysis and improved rates. In *International Conference on Machine Learning*, pages 5200–5209, 2019.

[GLSS18a] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1832–1841. PMLR, 10–15 Jul 2018.

[GLSS18b] Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[GLY+18] Yilong Geng, Shiyu Liu, Zi Yin, Ashish Naik, Balaji Prabhakar, Mendel Rosenblum, and Amin Vahdat. Exploiting a natural network effect for scalable, fine-grained clock synchronization. In *USENIX Symposium on Networked Systems Design and Implementation*, 2018.

[GR07] Olivier Guédon and Mark Rudelson. Lp-moments of random vectors via majorizing measures. *Advances in Mathematics*, 208(2):798–823, 2007.

[GRB+22] Eduard Gorbunov, Alexander Rogozin, Aleksandr Beznosikov, Darina Dvinskikh, and Alexander Gasnikov. *Recent Theoretical Advances in Decentralized Distributed Convex Optimization*, pages 253–325. Springer International Publishing, Cham, 2022.

[GWB+17] Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. *Advances in Neural Information Processing Systems*, 30, 2017.

[HAMS20] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey, 2020.

[HBM18] Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. Accelerated decentralized optimization with local updates for smooth and strongly convex objectives, 2018.

[HBM19] Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. An accelerated decentralized stochastic proximal algorithm for finite sums. *arXiv preprint arXiv:1905.11394*, 2019.

[HBM20] Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. Dual-free stochastic decentralized optimization with variance reduction. In *NeurIPS*, 2020.

[Hen22] Hadrien Hendrikx. A principled framework for the design and analysis of token algorithms, 2022.

[HHHR20] Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtarik. Lower Bounds and Optimal Algorithms for Personalized Federated Learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 2304–2315. Curran Associates, Inc., 2020.

[HHS17] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: Closing the generalization gap in large batch training of neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 1729–1739, 2017.

[HKP19] Christopher Hoffman, Matthew Kahle, and Elliot Paquette. Spectral gaps of random graphs and applications. *International Mathematics Research Notices*, 2021(11):8353–8404, May 2019.

[HKZ12] Daniel Hsu, Sham Kakade, and Tong Zhang. A tail inequality for quadratic forms of subgaussian random vectors. *Electronic Communications in Probability*, 17(none):1 – 6, 2012.

[HLA+21] Samuel Horváth, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos I. Venieris, and Nicholas D. Lane. FjORD: Fair and accurate federated learning under heterogeneous targets with ordered dropout. *Advances in Neural Information Processing Systems*, 34, 2021.

[HLT19] Fengxiang He, Tongliang Liu, and Dacheng Tao. Control batch size and learning rate to generalize well: Theoretical and empirical evidence. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[HMV15] Zhenqi Huang, Sayan Mitra, and Nitin Vaidya. Differentially Private Distributed Optimization. In *ICDCN*, 2015.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, January 1997.

[HW71] D. L. Hanson and F. T. Wright. A Bound on Tail Probabilities for Quadratic Forms in Independent Random Variables. *The Annals of Mathematical Statistics*, 42(3):1079 – 1083, 1971.

[HWLM21] Jeff Z. HaoChen, Colin Wei, Jason Lee, and Tengyu Ma. Shape matters: Understanding the implicit bias of the noise covariance. In *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134 of *Proceedings of Machine Learning Research*, pages 2315–2357. PMLR, 15–19 Aug 2021.

[HXB+20] Hadrien Hendrikx, Lin Xiao, Sebastien Bubeck, Francis Bach, and Laurent Massoulie. Statistically preconditioned accelerated gradient method for distributed optimization. 2020.

[IW14] Nobuyuki Ikeda and Shinzo Watanabe. *Stochastic differential equations and diffusion processes*. Elsevier, 2014.

[Jag13] Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 427–435, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[JGH18] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 8580–8589, 2018.

[JGS16] Pooria Joulani, András György, and Csaba Szepesvári. Delay-tolerant online convex optimization: Unified analysis and adaptive-gradient algorithms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[JKA+17] Stanisław Jastrzebski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Three factors influencing minima in SGD, 2017.

[JKA+18] Stanislaw Jastrzkebski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Width of minima reached by stochastic gradient descent is influenced by learning rate to batch size ratio. In *Artificial Neural Networks and Machine Learning – ICANN 2018*, pages 392–402, 2018.

[JKK+18] Prateek Jain, Sham M Kakade, Rahul Kidambi, Praneeth Netrapalli, and Aaron Sidford. Accelerating stochastic gradient descent for least squares regression. In *Conference On Learning Theory*, pages 545–604, 2018.

[JKRK19] Yihan Jiang, Jakub Konečný, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning, 2019.

[JP19] Yassir Jedra and Alexandre Proutiere. Sample complexity lower bounds for linear system identification. In *IEEE Conference on decision and control (CDC) 2019*, pages 2676–2681, 12 2019.

[JRJ07] Bjorn Johansson, Maben Rabi, and Mikael Johansson. A simple peer-to-peer algorithm for distributed optimization in sensor networks. In *2007 46th IEEE Conference on Decision and Control*, pages 4705–4710, 2007.

[JRJ10] Björn Johansson, Maben Rabi, and Mikael Johansson. A randomized incremental subgradient method for distributed optimization in networked systems. *SIAM Journal on Optimization*, 20(3):1157–1170, 2010.

[JS13] Jean Jacod and Albert Shiryaev. *Limit theorems for stochastic processes*, volume 288. Springer Science & Business Media, 2013.

[JT19] Ziwei Ji and Matus Telgarsky. Gradient descent aligns the layers of deep linear networks. In *International Conference on Learning Representations*, 2019.

[JT20] Ziwei Ji and Matus Telgarsky. Directional convergence and alignment in deep learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 17176–17186, 2020.

[JVB08] Laurent Jacob, Jean-philippe Vert, and Francis Bach. Clustered multi-task learning: A convex formulation. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2008.

[JWEG18] Bargav Jayaraman, Lingxiao Wang, David Evans, and Quanquan Gu. Distributed learning without distress: Privacy-preserving empirical risk minimization. In *NeurIPS*, 2018.

[KBB15] Walid Krichene, Alexandre Bayen, and Peter Bartlett. Accelerated mirror descent in continuous and discrete time. *Advances in Neural Information Processing Systems*, 28:2845–2853, 2015.

[KBT19] Mikhail Khodak, Maria-Florina F Balcan, and Ameet S Talwalkar. Adaptive Gradient-Based Meta-Learning Methods. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[Kel91a] F. P. Kelly. Loss networks. *The Annals of Applied Probability*, 1991.

[Kel91b] F. P. Kelly. Loss networks. *The Annals of Applied Probability*, 1(3):319–378, 1991.

[KF16] Donghwan Kim and Jeffrey A Fessler. Optimized first-order methods for smooth convex minimization. *Mathematical programming*, 159(1):81–107, 2016.

[KGGR21a] Dmitry Kovalev, Elnur Gasanov, Alexander Gasnikov, and Peter Richtárik. Lower bounds and optimal algorithms for smooth and strongly convex decentralized optimization over time-varying networks. In *NeurIPS*, 2021.

[KGGR21b] Dmitry Kovalev, Elnur Gasanov, Alexander Gasnikov, and Peter Richtarik. Lower bounds and optimal algorithms for smooth and strongly convex decentralized optimization over time-varying networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 22325–22335. Curran Associates, Inc., 2021.

[KKM+20] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5132–5143. PMLR, 13–18 Jul 2020.

[KKP20] Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. Survey of Personalization Techniques for Federated Learning. *arXiv:2003.08673 [cs, stat]*, March 2020. arXiv: 2003.08673.

[KL17] Vladimir Koltchinskii and Karim Lounici. Concentration inequalities and moment bounds for sample covariance operators. *Bernoulli*, 23(1):110 – 133, 2017.

[KLB+20] Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. A unified theory of decentralized sgd with changing topology and local updates. In *International Conference on Machine Learning*, pages 5381–5393. PMLR, 2020.

[Kle14] Achim Klenke. *The Poisson Point Process*. Springer London, 2014.

[KLN+08] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. What Can We Learn Privately? In *FOCS*, 2008.

[KLY18] Bobby Kleinberg, Yuanzhi Li, and Yang Yuan. An alternative view: When does SGD escape local minima? In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2698–2707. PMLR, 10–15 Jul 2018.

[KMA+19] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng

Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning, 2019.

[KMN+17a] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017.

[KMN+17b] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017.

[KMR20] Ahmed Khaled, Konstantin Mishchenko, and Peter Richtarik. Tighter theory for local sgd on identical and heterogeneous data. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 4519–4529. PMLR, 26–28 Aug 2020.

[KMRR16] Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated Optimization: Distributed Machine Learning for On-Device Intelligence. *arXiv:1610.02527 [cs]*, October 2016. arXiv: 1610.02527.

[KNJN21] Suhas Kowshik, Dheeraj Nagaraj, Prateek Jain, and Praneeth Netrapalli. Streaming linear system identification with reverse experience replay. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 30140–30152. Curran Associates, Inc., 2021.

[KR20] Ahmed Khaled and Peter Richtárik. Better theory for SGD in the nonconvex world. *arXiv Preprint arXiv:2002.03329*, 2020.

[KSJ19] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, volume 97, pages 3478–3487. PMLR, 2019.

[KSJ22] Anastasia Koloskova, Sebastian U. Stich, and Martin Jaggi. Sharper convergence guarantees for asynchronous SGD for distributed and federated learning. *arXiv preprint arXiv:2206.08307*, 2022.

[KSR20] Dmitry Kovalev, Adil Salim, and Peter Richtárik. Optimal and practical algorithms for smooth and strongly convex decentralized optimization. *NeurIPS*, 33, 2020.

[KSST09] Sham Kakade, Shai Shalev-Shwartz, and Ambuj Tewari. On the duality of strong convexity and strong smoothness: Learning applications and matrix regularization. *Unpublished Manuscript, http://ttic.uchicago.edu/shai/papers/KakadeShalevTewari09.pdf*, 2009.

[LACM13] Ji Liu, Brian D.O. Anderson, Ming Cao, and A. Stephen Morse. Analysis of accelerated gossip algorithms. *Automatica*, 49(4):873–883, 2013.

[Lat21] Jonas Latz. Analysis of stochastic gradient descent in continuous time. *Statistics and Computing*, 31(4):1–25, 2021.

[LCDW16] Jueyou Li, Guo Chen, Zhao Yang Dong, and Zhiyou Wu. Distributed mirror descent method for multi-agent optimization with delay. *Neurocomputing*, 2016.

[LG16] Jean-François Le Gall. *Brownian Motion, Martingales, and Stochastic Calculus*, volume 274. Springer, 2016.

[LHLL15] Xiangru Lian, Yijun Huang, Yuncheng Li, and Ji Liu. Asynchronous parallel stochastic gradient for nonconvex optimization. *Advances in Neural Information Processing Systems*, 28, 2015.

[LHZQ20] Qinyi Luo, Jiaao He, Youwei Zhuo, and Xuehai Qian. Prague: High-performance heterogeneity-aware asynchronous decentralized training. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020.

[LL19] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. *arXiv preprint arXiv:1906.05890*, 2019.

[LM12] Jure Leskovec and Julian Mcauley. Learning to discover social circles in ego networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[Lor66] G. G. Lorentz. Metric entropy and approximation. *Bulletin of the American Mathematical Society*, 72(6):903 – 937, 1966.

[LPLJ18] Rémi Leblond, Fabian Pedregosa, and Simon Lacoste-Julien. Improved asynchronous parallel optimization analysis for stochastic incremental methods. *Journal of Machine Learning Research*, 19:1–68, 2018.

[LPTG09] Karim Lounici, Massimiliano Pontil, AB Tsybakov, and SA Geer. Taking advantage of sparsity in multi-task learning. *Proceedings of the 22nd Conference on Information Theory*, 12 2009.

[LPW06] David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2006.

[LR18] Nicolas Loizou and Peter Richtárik. Accelerated gossip via stochastic heavy ball method, 2018.

[LRR19] Nicolas Loizou, Michael Rabbat, and Peter Richtárik. Provably accelerated randomized gossip algorithms. In *ICASSP 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 7505–7509. IEEE, 2019.

[LS07] Cassio G. Lopes and Ali H. Sayed. Incremental adaptive strategies over distributed networks. *IEEE Transactions on Signal Processing*, 55(8):4064–4077, 2007.

[LSBS20] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. In *International Conference on Learning Representations*, 2020.

[LSNK17] An-An Liu, Yu-Ting Su, Wei-Zhi Nie, and Mohan Kankanhalli. Hierarchical clustering multi-task learning for joint human action grouping and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(1):102–114, 2017.

[LWM19] Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards explaining the regularization effect of initial large learning rate in training neural networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.

[LYW+22] Qi Liu, Bo Yang, Zhaojian Wang, Dafeng Zhu, Xinyi Wang, Kai Ma, and Xinping Guan. Asynchronous decentralized federated learning for collaborative fault diagnosis of pv stations. *IEEE Transactions on Network Science and Engineering*, 9(3):1680–1696, 2022.

[LZZ+17] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems*, 30, 2017.

[LZZL18] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. Asynchronous decentralized parallel stochastic gradient descent. In *ICML*, 2018.

[Mas02] Laurent Massoulié. Stability of distributed congestion control with heterogeneous feedback delays. *IEEE Transactions on Automatic Control*, 2002.

[Mat88] Peter Matthews. Covering Problems for Markov Chains. *The Annals of Probability*, 16(3):1215 – 1228, 1988.

[MBEW22] Konstantin Mishchenko, Francis Bach, Mathieu Even, and Blake Woodworth. Asynchronous SGD beats minibatch SGD under arbitrary delays. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[MBM+16] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016.

[MHB16] Stephan Mandt, Matthew D. Hoffman, and David M. Blei. A variational analysis of stochastic gradient algorithms. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, page 354–363, 2016.

[MIMA18] Konstantin Mishchenko, Franck Iutzeler, Jérôme Malick, and Massih-Reza Amini. A delay-tolerant proximal-gradient algorithm for distributed learning. In *International Conference on Machine Learning*, pages 3584–3592, 2018.

[Min17] Stanislav Minsker. On some extensions of bernstein's inequality for self-adjoint operators. *arXiv preprint arXiv:1112.5448*, 2017.

[Mir17] Ilya Mironov. Renyi differential privacy. *CoRR*, abs/1702.07476, 2017.

[MJ19] Michael Muehlebach and Michael Jordan. A dynamical systems perspective on Nesterov acceleration. In *International Conference on Machine Learning*, pages 4656–4662. PMLR, 2019.

[MJPH21] Aritra Mitra, Rayana Jaafar, George J. Pappas, and Hamed Hassani. Linear convergence in federated learning: Tackling client heterogeneity and sparse gradients. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 14606–14619. Curran Associates, Inc., 2021.

[MKR20] Konstantin Mishchenko, Ahmed Khaled, and Peter Richtarik. Random reshuffling: Simple analysis with vast improvements. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 17309–17320. Curran Associates, Inc., 2020.

[ML18] Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks. *arXiv preprint arXiv:1804.07612*, 2018.

[MM10] Volodymyr Melnykov and Ranjan Maitra. Finite mixture models and model-based clustering. *Statistics Surveys*, 4(none):80 – 116, 2010.

[MMR+17] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[MMRS20] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three Approaches for Personalization with Applications to Federated Learning. *arXiv:2002.10619 [cs, stat]*, July 2020. arXiv: 2002.10619.

[MMS11] Eduardo Montijano, Juan Montijano, and C. Sagues. Chebyshev polynomials in distributed consensus applications. *IEEE Transactions on Signal Processing*, 61, 11 2011.

[MMS21] Rotem Mulayoff, Tomer Michaeli, and Daniel Soudry. The implicit bias of minima stability: A view from function space. In *Advances in Neural Information Processing Systems*, 2021.

[MNW+18] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. Ray: A distributed framework for emerging AI applications. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 561–577, 2018.

[MP67] V A Marčenko and L A Pastur. DISTRIBUTION OF EIGENVALUES FOR SOME SETS OF RANDOM MATRICES. *Mathematics of the USSR-Sbornik*, 1(4):457–483, apr 1967.

[MPP+17] Horia Mania, Xinghao Pan, Dimitris Papailiopoulos, Benjamin Recht, Kannan Ramchandran, and Michael I. Jordan. Perturbed iterate analysis for asynchronous stochastic optimization. *SIAM Journal on Optimization*, 27(4):2202–2229, 2017.

[MRTZ18] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning Differentially Private Recurrent Language Models. In *ICLR*, 2018.

[MS14] Brendan McMahan and Matthew J. Streeter. Delay-tolerant algorithms for asynchronous distributed online learning. *Advances in Neural Information Processing Systems*, 27, 2014.

[MSS19] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic Federated Learning. In *Proceedings of the 36th International Conference on Machine Learning*, pages 4615–4625. PMLR, May 2019. ISSN: 2640-3498.

[MW89] Bojan Mohar and Wolfgang Woess. A survey on spectra of infinite graphs. *Bulletin of the London Mathematical Society*, 21(3):209–234, 1989.

[MYH+20] Xianghui Mao, Kun Yuan, Yubin Hu, Yuantao Gu, Ali H. Sayed, and Wotao Yin. Walkman: A communication-efficient random-walk algorithm for decentralized optimization. *IEEE Transactions on Signal Processing*, 68:2513–2528, 2020.

[NBB01] Angelia Nedić, Dimitri P. Bertsekas, and Vivek S. Borkar. Distributed asynchronous incremental subgradient methods. *Studies in Computational Mathematics*, 8(C):381–407, 2001.

[NBO23] Adel Nabli, Eugene Belilovsky, and Edouard Oyallon. $\textbf{A}^2\textbf{CiD}^2$: Accelerating asynchronous communication in decentralized deep learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[NCTV19] Giovanni Neglia, Gianmarco Calbi, Don Towsley, and Gayane Vardoyan. The role of network topology for distributed machine learning. In *INFOCOM*, 2019.

[Nes83] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate o (1/k^ 2). 269:543–547, 1983.

[Nes03] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*, volume 87. Springer Science & Business Media, 2003.

[Nes12] Yurii Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

[Nes14] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition, 2014.

[Nic01] Silviu-Iulian Niculescu. *Delay effects on stability: a robust control approach*, volume 269. Springer Science & Business Media, 2001.

[NMZ+22] John Nguyen, Kshitiz Malik, Hongyuan Zhan, Ashkan Yousefpour, Mike Rabbat, Mani Malek, and Dzmitry Huba. Federated learning with buffered asynchronous aggregation. In *International Conference on Artificial Intelligence and Statistics*, pages 3581–3607, 2022.

[NO09] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

[NO23] Adel Nabli and Edouard Oyallon. DADAO: Decoupled accelerated decentralized asynchronous optimization. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 25604–25626. PMLR, 23–29 Jul 2023.

[NOR18] Angelia Nedich, Alex Olshevsky, and Michael G. Rabbat. Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106(5):953–976, May 2018.

[NRSS22] Mor Shpigel Nacson, Kavya Ravichandran, Nathan Srebro, and Daniel Soudry. Implicit bias of the step size in linear diagonal neural networks. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16270–16295. PMLR, 17–23 Jul 2022.

[NS17] Yurii Nesterov and Sebastian U. Stich. Efficiency of the accelerated coordinate descent method on structured optimization problems. *SIAM Journal on Optimization*, 27(1):110–123, 2017.

[NSB+15] Arun Nair, Praveen Srinivasan, Sam Blackwell, Cagdas Alcicek, Rory Fearon, Alessandro De Maria, Vedavyas Panneershelvam, Mustafa Suleyman, Charles Beattie, Stig Petersen, et al. Massively parallel methods for deep reinforcement learning. *arXiv preprint arXiv:1507.04296*, 2015.

[NSD+21] Giorgi Nadiradze, Amirmojtaba Sabour, Peter Davies, Shigang Li, and Dan Alistarh. Asynchronous decentralized SGD with quantized and local updates. *Advances in Neural Information Processing Systems*, 34, 2021.

[NTS14] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*, 2014.

[NWB+20] Dheeraj Nagaraj, Xian Wu, Guy Bresler, Prateek Jain, and Praneeth Netrapalli. Least squares regression with markovian data: Fundamental limits and algorithms. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16666–16676. Curran Associates, Inc., 2020.

[NY83] Arkadij Semenovič Nemirovskij and David Borisovich Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley-Interscience, 1983.

[Ope24] OpenAI. Sora: creating video from text: https://openai.com/sora, 2024.

[PF23] Scott Pesme and Nicolas Flammarion. Saddle-to-saddle dynamics in diagonal linear networks. *arXiv preprint arXiv:2304.00488*, 2023.

[PPVF21] S. Pesme, L. Pillaud-Vivien, and N. Flammarion. Implicit bias of SGD for diagonal linear networks: a provable benefit of stochasticity. In *Advances in Neural Information Processing Systems*, 2021.

[PRT22] Dario Pasquini, Mathilde Raynal, and Carmela Troncoso. On the privacy of decentralized machine learning. Technical report, arXiv:2205.08443, 2022.

[PVRF22] L. Pillaud-Vivien, J. Reygner, and N. Flammarion. Label noise (stochastic) gradient descent implicitly solves the lasso for quadratic parametrisation. In *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pages 2127–2159. PMLR, 2022.

[PVZ17] Grigoris Paouris, Petros Valettas, and Joel Zinn. Random version of Dvoretzky's theorem in lpn. *Stochastic Processes and their Applications*, 127(10):3187–3227, 2017.

[Rao12] Shravas Rao. Finding hitting times in various graphs, 2012.

[RGPP21] Max Ryabinin, Eduard Gorbunov, Vsevolod Plokhotnyuk, and Gennady Pekhimenko. Moshpit SGD: Communication-efficient decentralized training on heterogeneous unreliable devices. *Advances in Neural Information Processing Systems*, 34, 2021.

[RM51] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[RNV10] S. Sundhar Ram, A. Nedić, and V. V. Veeravalli. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of Optimization Theory and Applications*, 147(3):516–545, July 2010.

[RRWN11] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. *Advances in Neural Information Processing Systems*, 24, 2011.

[RT11] Angelika Rohde and Alexandre B. Tsybakov. Estimation of high-dimensional low-rank matrices. *The Annals of Statistics*, 39(2):887–930, 2011.

[Rus86] John Rust. Structural estimation of markov decision processes. In R. F. Engle and D. McFadden, editors, *Handbook of Econometrics*, volume 4, chapter 51, pages 3081–3143. Elsevier, 1 edition, 1986.

[RV13] Mark Rudelson and Roman Vershynin. Hanson-Wright inequality and sub-gaussian concentration. *Electronic Communications in Probability*, 18(none):1 – 9, 2013.

[SBB+17] Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal algorithms for smooth and strongly convex distributed optimization in networks. In *International Conference on Machine Learning*, 2017.

[SBB+19] Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Lee, and Laurent Massoulié. Optimal convergence rates for convex distributed optimization in networks. *Journal of Machine Learning Research*, 20:1–31, 2019.

[SBC14] Weijie Su, Stephen Boyd, and Emmanuel Candes. A differential equation for modeling Nesterov's accelerated gradient method: theory and insights. *Advances in neural information processing systems*, 27:2510–2518, 2014.

[SBR20] César Sabater, Aurélien Bellet, and Jan Ramon. Distributed Differentially Private Averaging with Improved Utility and Robustness to Malicious Parties. Technical report, arXiv:2006.07218, 2020.

[SCR+11] Elaine Shi, T.-H. Hubert Chan, Eleanor G. Rieffel, Richard Chow, and Dawn Song. Privacy-Preserving Aggregation of Time-Series Data. In *NDSS*, 2011.

[SDJS18] Bin Shi, Simon Du, Michael Jordan, and Weijie Su. Understanding the acceleration phenomenon via high-resolution differential equations. *arXiv preprint arXiv:1810.08907*, 2018.

[SDSJ19] Bin Shi, Simon Du, Weijie Su, and Michael Jordan. Acceleration via symplectic discretization of high-resolution differential equations. In *Advances in Neural Information Processing Systems*, volume 32, pages 5744–5752, 2019.

[SGF+10] Bharath K. Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert R.G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11(50):1517–1561, 2010.

[SHN+18] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *J. Mach. Learn. Res.*, 19(1):2822–2878, jan 2018.

[SK20] Sebastian U. Stich and Sai Praneeth Karimireddy. The error-feedback framework: Better rates for SGD with delayed gradients and compressed updates. *Journal of Machine Learning Research*, 21:1–36, 2020.

[SL18] Samuel L. Smith and Quoc V. Le. A Bayesian perspective on generalization and stochastic gradient descent. In *International Conference on Learning Representations*, 2018.

[SLRB17] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Math. Program.*, 162(1–2):83–112, mar 2017.

[SLW22] Tao Sun, Dongsheng Li, and Bao Wang. Adaptive random walk gradient descent for decentralized optimization. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 20790–20809. PMLR, 17–23 Jul 2022.

[SLWY15] Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. EXTRA: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

[SMS20] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems*, PP:1–13, 08 2020.

[SMT+18] Max Simchowitz, Horia Mania, Stephen Tu, Michael I. Jordan, and Benjamin Recht. Learning without mixing: Towards a sharp analysis of linear system identification. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pages 439–473. PMLR, 06–09 Jul 2018.

[Sna46] Severus Snape. *Advanced Potion Making, Snape's Personal Edition*. 1946.

[SSC+18] Natalie Stephenson, Emily Shane, Jessica Chase, Jason Rowland, David Ries, Nicola Justice, Jie Zhang, Leong Chan, and Renzhi Cao. Survey of machine learning techniques in drug discovery. *Current drug metabolism*, 19, 08 2018.

[SSR06] Nathan Srebro, Gregory Shakhnarovich, and Sam Roweis. An investigation of computational and informational limits in gaussian mixture clustering. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 865–872, New York, NY, USA, 2006. Association for Computing Machinery.

[SSY18] Tao Sun, Yuejiao Sun, and Wotao Yin. On markov chain gradient descent. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[SSZ14] Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *International Conference on Machine Learning*, pages 1000–1008. PMLR, 2014.

[SSZ20] Jesús María Sanz-Serna and Konstantinos Zygalakis. The connections between Lyapunov functions for some optimization algorithms and differential equations. *arXiv preprint arXiv:2009.00673*, 2020.

[Sti18] Sebastian U. Stich. Local SGD converges fast and communicates little. In *International Conference on Learning Representations*, 2018.

[SY18] Benjamin Sirb and Xiaojing Ye. Decentralized consensus algorithm with delayed and stochastic gradients. *SIAM Journal on Optimization*, 2018.

[SYLS16] Suvrit Sra, Adams Wei Yu, Mu Li, and Alexander J. Smola. Adadelay: Delay adaptive distributed stochastic optimization. In *Artificial Intelligence and Statistics*, pages 957–965. PMLR, 2016.

[T+15] Joel A Tropp et al. An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, 8(1-2):1–230, 2015.

[Tan95] Mike Tanner. *Practical queueing analysis*. IBM McGraw-Hill. McGraw-Hill, London, 1995.

[TBA86] John Tsitsiklis, Dimitri Bertsekas, and Michael Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE transactions on automatic control*, 31(9):803–812, 1986.

[Tc23] Hugo Touvron and coauthors. Llama: Open and efficient foundation language models, 2023.

[Tel13] Matus Telgarsky. Margins, shrinkage, and boosting. In *International Conference on Machine Learning*, pages 307–315. PMLR, 2013.

[TJJ20] Nilesh Tripuraneni, Michael Jordan, and Chi Jin. On the theory of transfer learning: The importance of task diversity. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7852–7862. Curran Associates, Inc., 2020.

[TJJ21] Nilesh Tripuraneni, Chi Jin, and Michael Jordan. Provable meta-learning of linear representations. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10434–10443. PMLR, 18–24 Jul 2021.

[TJNO21] Kiran Koshy Thekumparampil, Prateek Jain, Praneeth Netrapalli, and Sewoong Oh. Statistically and computationally efficient linear meta-representation learning. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[Tro11] Joel A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, Aug 2011.

[TSS20] Ye Tian, Ying Sun, and Gesualdo Scutari. Achieving linear convergence in distributed asynchronous multi-agent optimization. *IEEE Transactions on Automatic Control*, PP:1–1, 03 2020.

[TY09] Paul Tseng and Sangwoon Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117(1):387–423, 2009.

[Uhl94] Harald Uhlig. On singular wishart and singular multivariate beta distributions. *Ann. Statist.*, 22(1):395–405, 03 1994.

[VBS19] Sharan Vaswani, Francis Bach, and Mark Schmidt. Fast and faster convergence of sgd for over-parameterized models and an accelerated perceptron. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1195–1204. PMLR, 2019.

[VBT17] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. Decentralized Collaborative Learning of Personalized Models over Networks. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 509–517. PMLR, 20–22 Apr 2017.

[Ver11a] Roman Vershynin. Approximating the moments of marginals of high-dimensional distributions. *The Annals of Probability*, 39(4):1591 – 1606, 2011.

[Ver11b] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2011.

[Ver14] Roman Vershynin. Estimation in high dimensions: a geometric perspective. *arXiv preprint arXiv:1405.5103*, 2014.

[Ver18] Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge University Press, 2018.

[Ver20] Roman Vershynin. Concentration inequalities for random tensors. *arXiv preprint arXiv:1905.00802*, 2020.

[VKR19] Tomas Vaškevičius, Varun Kanade, and Patrick Rebeschini. Implicit regularization for optimal sparse recovery. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.

[VKR20] Tomas Vaskevicius, Varun Kanade, and Patrick Rebeschini. The statistical complexity of early-stopped mirror descent. In *Advances in Neural Information Processing Systems*, volume 33, pages 253–264, 2020.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[Wai19] Martin J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019.

[WB19] Jonathan Weed and Francis Bach. Sharp asymptotic and finite-sample rates of convergence of empirical measures in Wasserstein distance. *Bernoulli*, 25(4A):2620 – 2648, 2019.

[WCZT22] Yuqing Wang, Minshuo Chen, Tuo Zhao, and Molei Tao. Large learning rate tames homogeneity: Convergence and balancing effect. In *International Conference on Learning Representations*, 2022.

[WGL+20] Blake Woodworth, Suriya Gunasekar, Jason D. Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in overparametrized models. In Jacob Abernethy and Shivani Agarwal, editors, *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pages 3635–3673. PMLR, 09–12 Jul 2020.

[WGX18] Di Wang, Marco Gaboardi, and Jinhui Xu. Empirical Risk Minimization in Non-interactive Local Differential Privacy Revisited. In *NeurIPS*, 2018.

[Wis28] John Wishart. The generalised product moment distribution in samples from a normal multivariate population. *Biometrika*, 20A(1/2):32–52, 1928.

[WLHL15] Huiwei Wang, Xiaofeng Liao, Tingwen Huang, and Chaojie Li. Cooperative distributed optimization in multiagent networks with delays. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(2):363–369, 2015.

[WLMJ23] Xuyang Wu, Changxin Liu, Sindri Magnússon, and Mikael Johansson. Delay-agnostic asynchronous coordinate update algorithm. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[WLYZ22] Puyu Wang, Yunwen Lei, Yiming Ying, and Ding-Xuan Zhou. Stability and generalization for markov chain stochastic gradient methods. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[WME18] Lei Wu, Chao Ma, and Weinan E. How SGD selects the global minima in over-parameterized learning: A dynamical stability perspective. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[WMFJ22] Xuyang Wu, Sindri Magnusson, Hamid Reza Feyzmahdavian, and Mikael Johansson. Delay-adaptive step-sizes for asynchronous learning. *arXiv preprint arXiv:2202.08550*, 2022.

[WMK+19] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated Evaluation of On-device Personalization. *arXiv:1910.10252 [cs, stat]*, October 2019. arXiv: 1910.10252.

[Woj21] Stephan Wojtowytsch. Stochastic gradient descent with noise of machine learning type. part II: Continuous time analysis. *arXiv preprint arXiv:2106.02588*, 2021.

[WPS+20a] Blake Woodworth, Kumar Kshitij Patel, Sebastian Stich, Zhen Dai, Brian Bullins, Brendan Mcmahan, Ohad Shamir, and Nathan Srebro. Is local SGD better than minibatch SGD? In *International Conference on Machine Learning*, pages 10334–10343. PMLR, 2020.

[WPS20b] Blake E Woodworth, Kumar Kshitij Patel, and Nati Srebro. Minibatch vs local sgd for heterogeneous distributed learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6281–6292. Curran Associates, Inc., 2020.

[WR20] Fan Wu and Patrick Rebeschini. A continuous-time mirror descent approach to sparse phase retrieval. In *Advances in Neural Information Processing Systems*, volume 33, pages 20192–20203, 2020.

[Wri15] Stephen Wright. Coordinate descent algorithms. *Math. Program.*, 151(1, Ser. B):3–34, 2015.

[WRJ16] Ashia Wilson, Benjamin Recht, and Michael I Jordan. A Lyapunov analysis of momentum methods in optimization. *arXiv preprint arXiv:1611.02635*, 2016.

[WSY+19] Jianyu Wang, Anit Kumar Sahu, Zhouyi Yang, Gauri Joshi, and Soummya Kar. Matcha: Speeding up decentralized sgd via matching decomposition sampling. In *2019 Sixth Indian Control Conference (ICC)*, pages 299–300, 2019.

[WW23] Jun-Kun Wang and Andre Wibisono. Continuized acceleration for quasar convex functions in non-convex optimization. 2023.

[WWJ16] Andre Wibisono, Ashia C Wilson, and Michael I Jordan. A variational perspective on accelerated methods in optimization. *Proceedings of the National Academy of Sciences*, 113(47):E7351–E7358, 2016.

[WWS+18] Blake E. Woodworth, Jialei Wang, Adam Smith, Brendan McMahan, and Nati Srebro. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. *Advances in Neural Information Processing Systems*, 31, 2018.

[WYL+18] Tianyu Wu, Kun Yuan, Qing Ling, Wotao Yin, and Ali H. Sayed. Decentralized consensus optimization with asynchrony and delays. *IEEE Transactions on Signal and Information Processing over Networks*, 2018.

[WYX17] Di Wang, Minwei Ye, and Jinhui Xu. Differentially private empirical risk minimization revisited: Faster and more general. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NeurIPS*, volume 30. Curran Associates, Inc., 2017.

[WZBG21] Jingfeng Wu, Difan Zou, Vladimir Braverman, and Quanquan Gu. Direction matters: On the implicit bias of stochastic gradient descent with moderate learning rate. In *International Conference on Learning Representations*, 2021.

[XMX+18] Chenguang Xi, Van Sy Mai, Ran Xin, Eyad H. Abed, and Usman A. Khan. Linear convergence in optimization over directed graphs with row-stochastic matrices. *IEEE Transactions on Automatic Control*, 2018.

[XZW21] Jie Xu, Wei Zhang, and Fei Wang. A(dp)^{2}sgd: Asynchronous decentralized parallel stochastic gradient descent with differential privacy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[YBF23] Oğuz Yüksel, Etienne Boursier, and Nicolas Flammarion. First-order ANIL provably learns representations despite overparametrisation. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*, 2023.

[YBS21] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. Salvaging Federated Learning by Local Adaptation. *arXiv:2002.04758 [cs, stat]*, October 2021. arXiv: 2002.04758.

[YJY19] Hao Yu, Rong Jin, and Sen Yang. On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7184–7193. PMLR, 09–15 Jun 2019.

[YL16] Ke Ye and Lek-Heng Lim. Schubert varieties and distances between subspaces of different dimensions. *SIAM Journal on Matrix Analysis and Applications*, 37:1176–1197, 01 2016.

[YYC+21] Bicheng Ying, Kun Yuan, Yiming Chen, Hanbin Hu, Pan Pan, and Wotao Yin. Exponential graph is provably efficient for decentralized deep training. In *NeurIPS*, 2021.

[ZBH+17] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

[ZCY11] Jiayu Zhou, Jianhui Chen, and Jieping Ye. Clustered multi-task learning via alternating structure optimization. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, NIPS'11, page 702–710, Red Hook, NY, USA, 2011. Curran Associates Inc.

[ZDJW13] Yuchen Zhang, John Duchi, Michael I Jordan, and Martin J Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[Zei12] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[ZH20] Ligeng Zhu and Song Han. *Deep Leakage from Gradients*, page 17–31. Springer International Publishing, 2020.

[Zhi21] Nikita Zhivotovskiy. Dimension-free bounds for sums of independent matrices and simple tensors via the variational principle, 2021.

[Zho17] Yiqiao Zhong. Eigenvector under random perturbation: A nonasymptotic rayleigh-schrödinger theory. *arXiv preprint arXiv:1702.00139*, 2017.

[ZKL18] Xueru Zhang, Mohammad Mahdi Khalili, and Mingyan Liu. Improving the Privacy and Accuracy of ADMM-Based Distributed Algorithms. In *ICML*, 2018.

[ZMB+18] Zhengyuan Zhou, Panayotis Mertikopoulos, Nicholas Bambos, Peter Glynn, Yinyu Ye, Li-Jia Li, and Li Fei-Fei. Distributed asynchronous optimization with unbounded delays: How slow can you go? In *International Conference on Machine Learning*, pages 5970–5979. PMLR, 2018.

[ZMSJ18] Jingzhao Zhang, Aryan Mokhtari, Suvrit Sra, and Ali Jadbabaie. Direct Runge-Kutta discretization achieves acceleration. In *Advances in Neural Information Processing Systems*, volume 31, pages 3900–3909, 2018.

[ZMW17a] Kai Zheng, Wenlong Mou, and Liwei Wang. Collect at Once, Use Effectively: Making Non-interactive Locally Private Learning Possible. In *ICML*, 2017.

[ZMW+17b] Shuxin Zheng, Qi Meng, Taifeng Wang, Wei Chen, Nenghai Yu, Zhi-Ming Ma, and Tie-Yan Liu. Asynchronous stochastic gradient descent with delay compensation. In *International Conference on Machine Learning*, pages 4120–4129, 2017.

[ZWLS10] Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J. Smola. Parallelized stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 2595–2603, 2010.

[ZWW+23] Xingyu Zhu, Zixuan Wang, Xiang Wang, Mo Zhou, and Rong Ge. Understanding edge-of-stability training dynamics with a minimalist example. *International Conference on Learning Representations*, 2023.

[ZY21] Jiaqi Zhang and Keyou You. Fully asynchronous distributed optimization with linear convergence in directed networks, 2021.

RÉSUMÉ

Les modèles d'apprentissage automatique modernes exigent d'énormes quantités de données et de puissance de calcul pour effectuer des prédictions précises. Par conséquent, ces modèles sont entrainés de manière distribuée: un grand nombre d'unités de calcul sont nécessaires, et les données utilisées pour l'entraînement, étant trop vastes pour être centralisées sur une seule machine, sont stockées de manière décentralisée par les utilisateurs. Cette thèse explore différentes problématiques liées à l'entraînement distribué de modèles. En raison de l'échelle grandissante des ordres de grandeurs des besoins en calcul et en données nécessaires, nous examinons d'abord les méthodes d'entraînement asynchrones et le cadre décentralisé. Ensuite, nous abordons les questions liées à la protection des données, naturellement amplifiées dans le cadre décentralisé, et la personnalisation des modèles, un scénario plus complexe dans lequel les utilisateurs ne partagent pas nécessairement le même objectif mais doivent néanmoins collaborer.

MOTS CLÉS

Optimisation, statistiques, fédéré, distribué, collaboratif, décentralisé

ABSTRACT

The advent of modern Machine Learning (ML) models necessitates vast amounts of data and computational power to facilitate accurate predictions. Consequently, these models undergo training in a distributed manner, wherein numerous compute units are employed, and the voluminous training data, too extensive to be centralized on a single device, is decentralized across users' devices. This thesis delves into various aspects concerning the distributed training of models. Given the escalating scale of computational and data requirements, we initially concentrate on asynchronous training and decentralized methodologies, which confer robustness as the scale of the problem expands. Subsequently, we explore the inherent privacy amplification of decentralized learning and delve into model personalization—an intricate scenario wherein users may not share identical objectives yet necessitate collaboration.

KEYWORDS

Optimization, statistics, federated, distributed, collaborative, decentralized