Journal of Machine Learning Research 26 (2025) 1-48 Submitted 6/23; Revised 8/25; Published 10/25

Decentralized Asynchronous Optimization with DADAO allows Decoupling and Acceleration

Adel Nabli[1,2] ADEL.NABLI@SORBONNE-UNIVERSITE.FR

Edouard Oyallon[1] EDOUARD.OYALLON@CNRS.FR

[1] *Sorbonne University, CNRS, ISIR, Paris, France*

[2] *Mila, Concordia University, Montréal, Canada*

Editor: Ohad Shamir

Abstract

DADAO is the first decentralized, accelerated, asynchronous, primal, first-order algorithm to minimize a sum of L-smooth and μ-strongly convex functions distributed over a network of size n. Modeling the gradient updates and gossip communication procedures with separate independent Poisson Point Processes allows us to decouple the computation and communication steps, which can be run in parallel, while making the whole approach completely asynchronous. This leads to communication acceleration compared to synchronous approaches. Our method employs primal gradients and avoids using a multi-consensus inner loop and other ad-hoc mechanisms. By relating the smallest positive eigenvalue $1/\chi_1$ of the Laplacian matrix Λ and the maximal resistance $\chi_2 \leq \chi_1$ of the graph to a sufficient minimal communication rate, we show that DADAO requires $\mathcal{O}(n\sqrt{\frac{L}{\mu}}\log(\frac{1}{\epsilon}))$ local gradients and only $\mathcal{O}(\sqrt{\chi_1\chi_2}\,\mathrm{Tr}\,\Lambda\sqrt{\frac{L}{\mu}}\log(\frac{1}{\epsilon}))$ communications to reach ϵ-precision, up to logarithmic terms. Thus, we simultaneously obtain an accelerated rate for computations and communications, leading to an improvement over state-of-the-art works, our simulations further validating the strength of our relatively unconstrained method. Moreover, we propose a SDP relaxation to find the gossip rate of each edge minimizing the total number of communications for a given graph, resulting in faster convergence compared to standard approaches relying on uniform communication weights.

Keywords: Convex Optimization, Decentralized Methods, Asynchronous Optimization, Randomized Gossip, Distributed Computing

1. Introduction

In recent years, the increased amount of available data as well as the proliferation of highly-parallelizable and connected hardware have brought significant changes in the way we process data. These developments have led to the need for efficient and scalable methods for distributed optimization, particularly in the context of machine learning. Indeed, in scenarios where data is distributed across multiple nodes, such as in edge computing or distributed sensor networks, leveraging the local resources of each device is a topic of significant interest. In other settings, such as clusters, spreading the compute load is ideally done to obtain a linear speedup in the number of nodes. In a typical distributed training framework, the goal is to minimize a sum of functions $(f_i)_{i\leq n}$ split across n nodes of a computer network. A corresponding optimization procedure involves alternating local computations on the nodes and communications along the edges $\mathcal{E}$ of the network. In the decentralized setting, there is no central machine aggregating the information sent by the workers: nodes are only allowed to communicate with their neighbors in the network. This work addresses simultaneously multiple limitations of existing decentralized algorithms while guaranteeing fast convergence rates.

Synchronous lock. Optimal methods (Scaman et al., 2017; Kovalev et al., 2021a) have been derived for synchronous first-order algorithms, whose executions are blocked until all nodes have reached a predefined state (*e.g.*, they must all finish computing local gradients before the round of communication begins), which limits their efficiency in practice as they can heavily be impacted by a few slow nodes or edges in the graph (the *straggler problem*). To tackle the synchronous lock, we rely on the continuized framework (Even et al., 2021a), itself derived from the randomized gossip model (Boyd et al., 2006). In randomized gossip, nodes update their local values at random times using pairwise communication updates named gossip. Thus, iterates are randomized, labeled with a continuous-time index that only need local clocks to be synchronized at the beginning of the procedure (in opposition to a *global* iteration count that has to be known by all at all time) and performed locally with no regards to a specific global ordering of events. While based on discrete events and thus readily implementable, the continuized framework simplifies the analysis by leveraging continuous proof tools.

Coupled lock. However, in Even et al. (2021a), gradient and gossip operations are coupled: each communication along an edge first requires the computation of the gradients of the two functions locally stored on the corresponding nodes. As more communication steps than gradient computations are necessary to reach an ϵ precision, even in an optimal framework (Kovalev et al., 2021a; Scaman et al., 2017), the coupling leads to an overload in terms of gradient steps. Moreover, coupling computations and communications implies they must be performed *sequentially*, decoupling them allows both tasks to be done in *parallel*, allowing an additional speedup.

To our knowledge, our work is the first primal method to tackle those locks simultaneously while obtaining accelerated rates for both computations and communications. We propose a novel algorithm (DADAO: Decoupled Accelerated Decentralized Asynchronous Optimization) based on a combination of similar formulations to Kovalev et al. (2021a); Even et al. (2021b); Hendrikx (2022) in the continuized framework of Even et al. (2021a). We study:

$$\inf_{x\in\mathbb{R}^d} \sum_{i=1}^{n} f_i(x)\,, \tag{1}$$

where each $f_i : \mathbb{R}^d \to \mathbb{R}$ is a μ-strongly convex and L-smooth function computed in one of the n nodes of a network. We derive a first-order optimization algorithm that only uses primal gradients and relies on Point-wise Poisson Processes (P.P.P.s, see Last and Penrose (2017)) modeling of the communication and gradient occurrences, leading to accelerated communication and computation rates. Furthermore, our communication bounds rely on the maximal resistance of a graph rather than the largest eigenvalue of a Laplacian, leading to an additional acceleration compared to works which rely on synchrony. Our framework is based on a simple fixed-point iteration and kept minimal: it only involves primal computations with an additional momentum term. Thus, we do not add other cumbersome designs such as the Error-Feedback or Forward-Backward used in Kovalev et al. (2021a) (whose adaptation to asynchronous settings is for now unclear). While we do not consider the delays bound to appear in practice (we assume instantaneous communications and computations), we remove the coupling lock by performing gradient and gossip steps in parallel. Tab. 1 compares DADAO with other approaches and shows it is the only work to achieve accelerated rates both in number of communication and gradients. Then, we assume that each edge fires at a rate that can be adjusted *a priori*, yet the network has to verify a set of physical constraints, such as a maximal bandwidth condition. This contrasts with common gossip algorithms, which either leave the weights of the gossip matrix to be defined by the user or propose simple appropriate heuristics, such as the use of Metropolis weights, see *e.g.* Xiao et al. (2007); Shi et al. (2015); Nedic et al. (2017); Gorbunov et al. (2022). We cast a relaxation of the minimization of the communication rate as a novel convex optimization problem, which can be efficiently solved with classical solvers. This allows for outperforming communication rates obtained from uniform or standardized Laplacian weights, which are often suggested in the literature.

Difference with the ICML version. This journal version is an extension of the ICML paper Nabli and Oyallon (2023), in which we include the following points: **(1)** We introduce the problem of finding the optimal communication strategy for the *accelerated gossip* procedure considering various constraints. **(2)** We propose a relaxation of this problem and frame it as a semidefinite program (SDP). This approach allows us to determine the optimal gossip rates of each edge minimizing the total number of communications required for a given graph. **(3)** We find an analytical solution to the relaxed problem for the barbell graph and demonstrate that it leads to unboundedly better communication rates compared to using uniform communication weights. **(4)** We experimentally confirm that the edge weights found by solving the SDP does lead to accelerated rates of convergence in practice when running the accelerated gossip algorithm compared to the naive strategy. **(5)** We discuss the relaxation technique employed and its connection to the Braess paradox.

Contributions. **(1)** We propose a primal algorithm with provable guarantees in the context of asynchronous decentralized learning. **(2)** This algorithm is the first to reach accelerated rates for both communications and computations while not requiring ad-hoc mechanisms obtained from an inner loop. **(3)** We propose a simple theoretical framework compared to concurrent works, we show that our rates are better than previous works, and **(4)** we illustrate this theoretical comparison numerically. **(5)** We propose a semi-definite program (SDP) procedure to optimize the communication rate of our algorithm for a user-specified graph, and we demonstrate its advantage on the Barbell graph.

Structure of the paper. In Sec. 3.1, we describe our work hypothesis and our model of a decentralized environment, while Sec. 3.2 describes our dynamic. Sec. 3.3 states our convergence guarantees and highlights that the communication and computational rates of our method are better compared to its competitors. Next, Sec. 4 explains how to optimize the edge weights of a given graph topology to minimize DADAO's overall communication rate. Sec. 5.1 explains our implementation of this algorithm, and finally, Sec. 5.2 verifies our claims numerically. All our experiments are reproducible, using PyTorch (Paszke et al., 2019), our code being online https://github.com/AdelNabli/DADAO/.

Notations: $f = \mathcal{O}(g)$ means there is a constant $C > 0$ such that $|f| \leq C|g|$, $\{e_i\}_{i \leq d}$ is the canonical basis of $\mathbb{R}^d, d \in \mathbb{N}$, $\mathbf{1}$ is the vector of 1, $\mathbf{I}$ the identity, A^+ is the pseudo-inverse of A. We further write $\mathbf{e}_i \triangleq e_i \otimes \mathbf{I}$.

2. Related Work

Table 1: This table shows the strength of DADAO compared to concurrent works for obtaining ϵ-precision. n is the number of nodes, $|\mathcal{E}|$ the number of edges, $\frac{1}{\chi_1}$ the smallest positive eigenvalue of a fixed weighted Laplacian $\mathcal{L}$ so that $\operatorname{Tr}\mathcal{L} = n$, ρ the eigengap and $\chi_2 \leq \chi_1$ the effective resistance. Note that under reasonable assumptions $\sqrt{\chi_1\chi_2}n = \mathcal{O}(|\mathcal{E}|\sqrt{\rho})$ (see Lemma 2). Async., Comm., Grad., M.-C. and Prox. stand respectively for Asynchrony, Communication steps, Gradient steps., Multi-consensus and Proximal operator. As suggested in their respective papers, all the concurrent algorithms are run with $\frac{1}{\|\mathcal{L}\|}\mathcal{L}$. For AGT and OGT, the mixing matrix used is stochastic, a more precise comparison is given by Prop. 5.

Method	Async.	Decoupled	No Inner Loop (M.-C. or Prox.)	Primal Oracle	Total # Comm.	Total # Grad.
MSDA (Scaman et al., 2017)	✗	✗	✗	✗	$\sqrt{\rho}\|\mathcal{E}\|\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$	$n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$
DVR (Hendrikx et al., 2020)	✗	✗	✗	✓	$\sqrt{\rho}\|\mathcal{E}\|\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$	$n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$
ADOM+ (Kovalev et al., 2021a)	✗	✗	✗	✓	$\rho\|\mathcal{E}\|\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$	$n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$
TVR (Hendrikx, 2022)	✗	✓	✗	✓	$\rho\|\mathcal{E}\|\frac{L}{\mu}\log\frac{1}{\epsilon}$	$n\frac{L}{\mu}\log\frac{1}{\epsilon}$
AGT (Li and Lin, 2021)	✗	✗	✗	✓	$\sqrt{\rho}\|\mathcal{E}\|\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$	$n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$
OGT (Song et al., 2021)	✗	✓	✓	✓	$\sqrt{\rho}\|\mathcal{E}\|\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$	$n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$
ESDACD (Hendrikx et al., 2019b)	✓	✗	✓	✗	$\sqrt{\chi_1\chi_2}n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$	$\sqrt{\chi_1\chi_2}n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$
Continuized (Even et al., 2021a)	✓	✗	✓	✗	$\sqrt{\chi_1\chi_2}n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$	$\sqrt{\chi_1\chi_2}n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$
DADAO (ours)	✓	✓	✓	✓	$\sqrt{\chi_1\chi_2}n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$	$n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}$

Continuized and asynchronous algorithms. We highly rely on the elegant continuized framework (Even et al., 2021a), which allows obtaining simpler proofs and brings the flexibility of asynchronous algorithms. We reemphasize that identically to Even et al. (2021a), the result of this paper is *a stochastic discrete algorithm with a continuous proof:* our proof framework is not based on the discretization of an Ordinary Differential Equation (ODE) but rather studies *the evolution of a Stochastic Differential Equation (SDE) with jumps.* However, by contrast to Even et al. (2021a), in our work, we significantly reduce the necessary amount of gradient steps compared to Even et al. (2021a) while maintaining the same amount of activated edges. Another type of asynchronous algorithm can also be found in Latz (2021), yet it fails to obtain Nesterov's accelerated rates for lack of momentum. We note that Leblond et al. (2018) studies the robustness to delays yet requires a shared memory and thus applies to a different context than decentralized optimization. Hendrikx (2022) is a promising approach for modeling random communication on graphs yet fails to obtain acceleration in a neat framework without inner loops.

Decentralized algorithms with fixed topology. Scaman et al. (2017) is the first work to derive an accelerated algorithm for decentralized optimization, and it links the convergence speed to the Laplacian eigengap. The corresponding algorithm uses a dual formulation and a Chebychev acceleration (synchronous and only for fixed topology). Yet, as stated in Tab. 2, it still requires many edges to be activated. Furthermore, under a relatively flexible condition on the intensity of our P.P.P.s, we show that our work improves over bounds that depend on the spectral gap. An emerging line of work following this formulation employs the continuized framework (Even et al., 2020, 2021a,b), but unfortunately do not use a primal oracle, as they rely on the gradients of the Fenchel conjugate. Finally, we note that the work of Even et al. (2021b) incorporates delays in their model, showing that, with some adaptation, the continuized methods in Even et al. (2021a) still converge at a linear rate. Yet transferring this robustness to our setting remains unclear. Reducing the number of communication has been studied in Mishchenko et al. (2022), but without obtaining accelerated rates. Hendrikx et al. (2021) allows for fast communication and gossip rates yet requires a proximal step and synchrony between nodes to apply a momentum variable.

Finite sum acceleration. If each local function f_i is a sum of elementary functions $\sum_{j=1}^{m} f_{i,j}$ with a favorable conditionning, an additional acceleration is possible, as observed by Hendrikx et al. (2021, 2020); Hendrikx (2022), which is a different problem from Eq. 1. This can be viewed as a cluster of nodes with infinite connectivity and an efficient decentralized algorithm should automatically adapt to such structure. Thus, we focused on the setting $m = 1$, which allows a fair comparison with these works.

Error feedback/Gradient tracking. A major lock for asynchrony is the use of Gradient Tracking (GT) (Koloskova et al., 2021; Nedic et al., 2017; Li and Lin, 2021) or Error Feedback (Stich and Karimireddy, 2020; Kovalev et al., 2021b). Indeed, gradient operations are locally tracked by a running-mean variable which must be synchronously updated at each gradient update (in GT, the sum of this distributed variable keeps track of the gradient of the objective function), making it incompatible with an asynchronous framework. Zhang and You (2019) uses a GT-*like* procedure, which do not verify this property, at the cost of using a buffer (increasing memory requirements) and worse convergence rates (*e.g.*, not accelerated). Furthermore, acceleration requires an undesirable multi-consensus inner loop. We emphasize that Song et al. (2021) allows to decouple the gradient updates from communication, yet the framework is still synchronous, leading to synchronous communication rates, that asynchrony can improve (see Tab. 1).

Decoupling procedures. Decoupling subsequent steps of optimization procedures traditionally leads to speed-ups (Hendrikx et al., 2021; Hendrikx, 2022; Belilovsky et al., 2020, 2021). This contrasts with methods which couple gradient and gossip updates, so that they happen in a predefined order, i.e., simultaneously (Even et al., 2021a) or sequentially (Kovalev et al., 2021a; Koloskova et al., 2020). In decoupled optimization procedures, inner-loops are not desirable as they require an external procedure that can be potentially slow and need a block-barrier instruction during the algorithm's execution (e.g., Hendrikx et al. (2021); Nabli et al. (2025)). It means in particular that it is preferrable to avoid approaches such as Catalyst (Lin et al., 2015), multi-consensus steps (Kovalev et al., 2021c) or Tchebychev acceleration of consensus (Scaman et al., 2017).

Resistance of a graph. The maximal resistance of a graph is a widely studied quantity, particularly in physics (Klein and Randić, 1993; Vos, 2016; Klein, 2002), as it is a refined geometric invariant of graphs. The resistance of a graph corresponds to the commute time of a Markov Chain (Chandra et al., 1996). However, beyond the Continuized framework Even et al. (2021a) or acceleration of consensus problems (Can et al., 2022; Aybat and Gürbüzbalaban, 2017; Ghosh et al., 2008; Nabli et al., 2023), we are unaware of generic, asynchronous, accelerated decentralized optimization procedures that rely on this quantity. Also, it has a more physical interpretation than the Laplacian's norm, as it can be computed via Ohm's and Kirchhoff's Circuit Laws (see Chandra et al. (1996)).

Network optimization for gossip. Optimizing graph weights to reduce the communication or improving the convergence rate of a decentralized algorithm is an active area of research, for instance to reduce the variance between workers in the context of stochastic optimization (Le Bars et al., 2022; Vogels et al., 2022). Another line of works proposes to design graphs with high-connectivity properties (Lubotzky, 2012; Chow et al., 2016), referred to as *graph expanders*. However, this requires flexible network constraints. Note that our weight optimization procedure could still be run in an ad-hoc manner on those sub-graphs to optimize their communication rates. Given a graph, Xiao and Boyd (2003); Boyd et al. (2004) cast as a SDP the search for edge weights minimizing the convergence time of the standard gossip algorithm, showing that an optimal averaging algorithm can perform arbitrarily better than the one based on the Metropolis weights on common graphs, such as the Barbell graph. However, their study only focuses on the *non-accelerated* gossip algorithm, and extending their work to find faster gossip matrices for accelerated gossip remains to be done. Finally, other work propose to sparsify a graph with heuristic algorithms (Tang et al., 2020).

3. Accelerated Asynchronous Algorithm

3.1 Gossip Framework

We consider the problem defined by Eq. (1) in a distributed environment constituted by n nodes whose dynamic is indexed by a continuous time index $t \in \mathbb{R}^+$. Each node has a local memory and can compute a local gradient ∇f_i, as well as elementary operations, in an instantaneous manner. As said above, having no delay is less realistic, yet adding them also leads to significantly more difficult proofs whose adaptation to our framework remains largely unclear. Next, we will assume that our computations and gossip result from independent homogeneous P.P.P. with no delay. For the sake of simplicity, we assume that all nodes can compute a gradient at the same rate:

Assumption 3.1 (Homogeneous gradient computations) *The gradient computations are normalized to fire independently at a rate of 1 computation per time unit. For the i-th worker, we write $N_i(t)$ the corresponding P.P.P. of rate* 1*, as well as* $\mathbf{N}(t) = (N_i(t))_{i\leq n}$.

Remark 1 *The P.P.P $N_i(t)$ on node i means that taking gradient steps at i are discrete events, but the time intervals between two events is a random variable following an exponential law of parameter* 1*. Thus, the expected waiting time between two gradient steps on the i-th worker is* 1 *time unit.*

Next, we model the bandwidth of each connection. For an edge (i,j) belonging to $\in \mathcal{E}$, the set of edges of a graph we assume connected (see Eq. (3.2)), we write $M_{ij}(t)$ the P.P.P. with rate $0 < \lambda_{ij} < \infty$. When this P.P.P. fires, both nodes share and update their local memories. The rate λ_{ij} is adjustable locally by machine i while λ_{ji} is controlled by machine j. While λ_{ij} and λ_{ji} may be different, we highlight that the communication process is symmetric, *i.e.* both nodes update their local memories when either one of the two corresponding P.P.Ps fires. Thus, in the corresponding undirected graph $\bar{\mathcal{E}}$, each edge (i,j) will fire at a rate of $\lambda_{ij}+\lambda_{ji}$. Given our notations, if $(i,j)\notin\mathcal{E}$, then the connection between (i,j) can be thought as a P.P.P. with intensity 0. Taking the λ_{ij} as edge weights, we introduce the subsequent graph Laplacian, which is the expected Laplacian of our graph:

$$\Lambda \triangleq \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}(e_i-e_j)(e_i-e_j)^{\mathsf{T}}\,.$$

We write $\mathbf{\Lambda} \triangleq \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^{\mathsf{T}}$ its tensorized counter-part that will be useful for our Lyapunov-based proofs. Following Scaman et al. (2017), we will compare this quantity to the following projector:

$$\pi \triangleq \mathbf{I}-\frac{1}{n}\mathbf{1}\mathbf{1}^{\mathsf{T}} = \frac{1}{2n}\sum_{1\le i,j\le n}(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^{\mathsf{T}}\,.$$

In this context, a natural quantity is the algebraic connectivity of our network given by Kovalev et al. (2021a) as $\chi_1 \triangleq \sup_{\|x\|=1, x\perp\mathbf{1}} \frac{1}{x^{\mathsf{T}}\Lambda x}$. We might also write $\chi_1[\Lambda]$ to avoid confusion, depending on the context.

Assumption 3.2 (Connected graph) *The set of edges $\mathcal{E}$ defines a connected graph such that $\chi_1[\Lambda]<\infty$.*

Next, the maximal effective resistance of the network, as in Even et al. (2021a); Ellens et al. (2011), is:

$$\chi_2 \triangleq \frac{1}{2}\sup_{(i,j)\in\mathcal{E}} (e_i-e_j)^{\mathsf{T}}\Lambda^{+}(e_i-e_j)\,.$$

A standard quantity (Scaman et al., 2017), which is used to control the number of synchronous gossips steps, is the spectral gap, given by $\rho \triangleq \|\Lambda\|\chi_1$. We also introduce the value κ, the ratio of communication frequency between the fastest and slowest edges:

$$\kappa \triangleq \frac{\sup_{(i,j)\in\mathcal{E}}\lambda_{ij}+\lambda_{ji}}{\inf_{(i,j)\in\mathcal{E}}\lambda_{ij}+\lambda_{ji}}\,.$$

This ratio is typically bounded, for instance, in the case of a graph with constant edge weights or for $\lambda_{ij}=\frac{1}{d_i}$ with d_i the degree of the i-th node in a bounded degree graph or a regular graph. We prove the following Lemma (proved in Appendix A), which is useful to control χ_1,χ_2 and compare our bounds with works that rely on the spectral gap of a graph:

Lemma 2 (Effective resistance) *The spectrum of Λ is non-negative. Also, we have $\chi_1 = +\infty$ iff $\bar{\mathcal{E}}$ is not a connected graph. Also, if the graph is connected, then:*

$$\frac{n-1}{\operatorname{Tr}\Lambda} \le \chi_2 \le \chi_1\,.$$

Furthermore, we also have the following:

$$\sqrt{\chi_1\chi_2}\mathrm{Tr}\,\Lambda \leq \sqrt{\rho}\sqrt{\kappa n|\bar{\mathcal{E}}|}\,.$$

The last part of this Lemma indicates that our method requires less communications than methods depending on the spectral gap when no degenerated behavior on the graph's connectivity happens, *i.e.* when κ is adequately bounded, which is a standard assumption (Hendrikx et al., 2019a) as edges are usually weighted by their degree which is no more than n.

Remark 3 *For synchronous frameworks (Kovalev et al., 2021a; Scaman et al., 2017), the spectral quantity extracted from their gossip matrix is a given measure of the connectedness of their graphs that is used afterwards to deduce the right number of synchronous communication rounds between two gradient steps. In our framework, Λ directly contains the information of both the topology $\mathcal{E}$ and the edge communication rates λ_{ij}, thus $\chi_1[\Lambda]$ must rather be understood as an indicator of how well the graphs connectedness and the chosen communication strategy interact. In fact, we will see later that there is a condition on χ_1, χ_2 for DADAO to converge, see Appendix G for further discussion.*

3.2 Dynamic to optimum

Next, we follow a standard approach (Kovalev et al., 2020, 2021a; Salim et al., 2022; Hendrikx, 2022) for solving Eq. 1 (see Appendix B for details), leading to studying, for $0 < \nu < \mu$, the following Lagrangian:

$$\inf_{x\in\mathbb{R}^{n\times d}} \sup_{\substack{y\in\mathbb{R}^{n\times d}\\ z\in\mathbb{R}^{n\times d}}} \sum_{i=1}^{n} f_i(x_i) - \frac{\nu}{2}\|x\|^2 - \langle x, y\rangle - \frac{1}{2\nu}\|\pi z + y\|^2.$$

For $f(x) = \sum_{i=1}^n f_i(x_i)$, the saddle points (x^*, y^*, z^*) of the above Lagrangian are given by:

$$\begin{cases} \nabla f(x^*) - \nu x^* - y^* & = 0 \\ y^* + \pi z^* + \nu x^* & = 0 \\ \pi z^* + \pi y^* & = 0\,. \end{cases} \tag{2}$$

Our algorithm is based on a fixed-point algorithm to obtain those saddle points, which is a similar idea to Kovalev et al. (2021b), which is only restricted to a setting without communication acceleration. Furthermore, contrary to Kovalev et al. (2021a), we do not employ a Forward-Backward algorithm, which requires both an extra-inversion step and additional regularity on the considered proximal operator. Not only does this condition not hold in this particular case, but this is not desirable in a continuized framework where iterates are not ordered in a predefined sequence and require a local descent at each instant. Another major difference is that no Error-feedback is required by our approach, which allows unlocking asynchrony while simplifying the proofs and decreasing the required number of communications. Instead, we show it is enough to incorporate a standard fixed point algorithm, *without any specific preconditioning* (see Condat et al. (2019)). We consider the following dynamic:

$$\begin{aligned}
dx_t &= \eta(\tilde{x}_t - x_t)dt - \gamma(\nabla f(x_t) - \nu x_t - \tilde{y}_t)\, d\mathbf{N}(t) \\
d\tilde{x}_t &= \tilde{\eta}(x_t - \tilde{x}_t)dt - \tilde{\gamma}(\nabla f(x_t) - \nu x_t - \tilde{y}_t)\, d\mathbf{N}(t) \\
d\tilde{y}_t &= -\theta(y_t + z_t + \nu \tilde{x}_t)dt + (\delta + \tilde{\delta})(\nabla f(x_t) - \nu x_t - \tilde{y}_t) d\mathbf{N}(t) \\
dy_t &= \alpha(\tilde{y}_t - y_t)dt \\
dz_t &= \alpha(\tilde{z}_t - z_t)dt - \beta \sum_{(i,j)\in\mathcal{E}} (\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^{\mathsf{T}}(y_t + z_t) dM_{ij}(t) \\
d\tilde{z}_t &= \tilde{\alpha}(z_t - \tilde{z}_t)dt - \tilde{\beta} \sum_{(i,j)\in\mathcal{E}} (\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^{\mathsf{T}}(y_t + z_t) dM_{ij}(t)\,,
\end{aligned} \tag{3}$$

where $\nu, \tilde{\eta}, \eta, \gamma, \alpha, \tilde{\alpha}, \theta, \delta, \tilde{\delta}, \beta, \tilde{\beta}$ are undetermined real-valued parameters. As in Nesterov (2003), variables $(x_t, \tilde{x}_t)$, $(y_t, \tilde{y}_t)$ and $(z_t, \tilde{z}_t)$ are paired to obtain a Nesterov acceleration. The variables (x, y) allow decoupling the gossip steps from the gradient steps using independent P.P.P.s. Furthermore, the Lebesgue integrable path of $\tilde{y}_t$ does not correspond to a standard momentum, as in a continuized framework (Even et al., 2021a); however, it turns out to be a crucial component of our method. Compared to Kovalev et al. (2021a), no extra multi-consensus step needs to be integrated. Our formulation of an asynchronous gossip step is similar to Even et al. (2021a), which introduces a stochastic variable on edges; however, contrary to this work, our gossip and gradient computations are decoupled. We emphasize that while the dynamic 3 is formulated using SDEs (Arnold, 1974), which brings the power of the continuous-time analysis toolbox, it is still *event-based* and thus discrete in nature. Hence, the dynamic can be efficiently implemented in practice as explained in Sec. 5.1 and Appendix H.

3.3 Theoretical guarantees

We prove the following in Appendix C.

Theorem 4 *Assume each f_i is μ-strongly convex and L-smooth. Assume 3.1, then there exists some parameters for the dynamic Eq.* (3) *(given in Lemma 12), such that for any initialization $x_0 \in ker(\pi)$, and $\tilde{x}_0 = x_0, y_0 = \tilde{y}_0 = \nabla f(x_0) - \frac{\mu}{2}x_0, z_0 = \tilde{z}_0 = -\pi y_0$, we get for $t \in \mathbb{R}^+$:*

$$\mathbb{E}[\|x_t - x^*\|^2] \leq 4(1 + \frac{L}{\mu} + \frac{L^2}{\mu^2})\|x_0 - x^*\|^2 e^{-\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}}$$

Also, the expected number of oracle gradient calls is nt and the expected number of edges activated is: $\frac{t}{2}\operatorname{Tr}\Lambda$.

Discussion on hyperparameters and Laplacian tuning. Appendix C.1 shows that the parameters depend on the problem constants μ, L, χ_1, χ_2, implying they must be known in advance. Consequently, the method is not adaptive, particularly to communication constraints (as in Kovalev et al. (2021c)). Naively, the algorithm requires $\mathcal{O}\left(n\sqrt{\chi_1\chi_2\frac{L}{\mu}}\right)$ iterations and $\mathcal{O}\left(\sqrt{\chi_1\chi_2\frac{L}{\mu}}\operatorname{Tr}\Lambda\right)$ communications. To achieve the rates in Table 1, we need to introduce a scaled Laplacian

$$\tilde{\Lambda} = \sqrt{\chi_1[\Lambda]\chi_2[\Lambda]}\,\Lambda\,. \tag{4}$$

This rescaling ensures $\chi_1[\tilde{\Lambda}]\chi_2[\tilde{\Lambda}] = 1$ and preserves the product $\sqrt{\chi_1[\Lambda]\chi_2[\Lambda]}\,\mathrm{Tr}\,\Lambda$, yielding the desired rates. (see Appendix G for details).

This property allows us to use in DADAO the Laplacians introduced in previous work, to which we can now compare. It leads to the following proposition (proved in Appendix E), which shows that DADAO obtains better complexities than concurrent works while starting from the same family of Laplacians:

Proposition 5 (Comparison with concurrent work)

- *If $\kappa = \mathcal{O}(\frac{|\bar{\mathcal{E}}|}{n})$, then DADAO obtains better communication rate than MSDA (Scaman et al., 2017), and requires strictly less communications for the complete graph.*
- *For any fixed Laplacian valid for ADOM+ (Kovalev et al., 2021a), DADAO obtains a better communication rate than ADOM+, and requires strictly less communications for the complete graph.*
- *If $\kappa = \mathcal{O}(\frac{|\bar{\mathcal{E}}|}{n})$, for a valid Laplacian using the Gossip matrix of OGT (Song et al., 2021) or AGT (Li and Lin, 2021), DADAO obtains a better communication rate than both Gradient Tracking methods, and requires strictly less communications for the complete graph.*
- *DADAO requires fewer gradient computations than the Continuized framework (Even et al., 2021b), and has a strictly better computation rate for the cycle graph.*

We highlight that Scaman et al. (2017) claimed that their algorithm is optimal because they study the number of computations and *synchronized* gossips on a worst-case graph; our claim is, *by nature* different, as we are interested in the number of edges fired rather than the number of synchronized gossip rounds. Indeed, in an asynchronous framework, there is no notion of *round of* communication, which allows this framework to enjoy the advantageous rates of randomized procedures. Moreover, not only this measure of complexity is standard in asynchronous frameworks (*e.g.*, see Boyd et al. (2006)), but also, if we are aiming for the most frugal procedure, minimizing both the total number of computations and communications is of interest. Tab. 2 predicts the behavior of our algorithm on various classes of graphs encoded via a normalized Laplacian. It shows that systematically, our algorithm leads to the best complexities. For example, in the case of a complete graph, one synchronized gossip round requires a total of $|\mathcal{E}| = \mathcal{O}(n^2)$ communications whereas we show that a rate of only $\mathcal{O}(n)$ communications per time unit suffices for DADAO in this case. We note that the graph class depicted in Tab. 2 was used as worst-case examples for proving the optimality of Scaman et al. (2017) in a synchronous context.

4. Minimizing the communication rate as a SDP

For a given connected graph topology $\mathcal{E}$, we introduce $\mathbb{R}_+^{\mathcal{E}} \triangleq \{\lambda, \lambda_{ij} \geq 0 \text{ and } \lambda_{ij} = 0, \text{ if } (i,j) \notin \mathcal{E}\}$. We want to find a set of edge firing rates that would minimize the overall number of

Table 2: Complexity for various graphs using $\frac{1}{\|\mathcal{L}\|}\mathcal{L}$ with $\mathcal{L}$ the standard Laplacian with unit edge weights. In this case, $\rho = \chi_1$. The complexities are reported per time unit so that all algorithms reach ϵ-precision at the same time. We have, respectively, for a star/line or cyclic/complete graph and the d-dimensional grid, computed the exact rates in Appendix C.1. In the centralized setting, we assume that each edge is activated twice in succession: once to communicate gradients, and once to transmit the updated parameters.

Method	# edges activated per time unit				# gradients computed per time unit			
Graph	Star	Line	Complete	d-grid	Star	Line	Complete	d-grid
(Kovalev et al., 2021a) ADOM+	n^2	$8\pi^{-1}n^3$	n^2	$4\pi^{-1}d^2n^{1+2/d}$	n	n	n	n
(Scaman et al., 2017) MSDA	$n^{\frac{3}{2}}$	$4\pi^{-\frac{1}{2}}n^2$	n^2	$2\pi^{-\frac{1}{2}}d^{\frac{3}{2}}n^{1+1/d}$	n	n	n	n
(Even et al., 2021a) Continuized	$\sqrt{2}n$	$\sqrt{2}\pi^{-\frac{1}{2}}n^2$	n	$\sqrt{2}\pi^{-\frac{1}{2}}d^{\frac{1}{2}}n^{1+1/d}$	$\sqrt{2}n$	$\sqrt{2}\pi^{-\frac{1}{2}}n^2$	n	$2\pi^{-\frac{1}{2}}d^{\frac{1}{2}}n^{1+1/d}$
Centralized	$2n$	–	–	–	n	–	–	–
DADAO (ours)	$\sqrt{2}n$	$\sqrt{2}\pi^{-\frac{1}{2}}n^2$	n	$\sqrt{2}\pi^{-\frac{1}{2}}d^{\frac{1}{2}}n^{1+1/d}$	n	n	n	n

communications. As said in Sec.3.3, given a set of weights $\lambda \in \mathbb{R}_+^{\mathcal{E}}$ defining a Laplacian $\Lambda(\lambda)$, it is sufficient to use the edge firing rates given by the normalized Laplacian $\Lambda(\lambda)\sqrt{2\chi_1[\Lambda(\lambda)]\chi_2[\Lambda(\lambda)]}$ to guarantee convergence. Thus, according to Th.4, the quantity we want to minimize is the trace of this normalized Laplacian, *i.e.* $\operatorname{Tr}\Lambda(\lambda)\sqrt{2\chi_1[\Lambda(\lambda)]\chi_2[\Lambda(\lambda)]}$. However, except in a simplistic scenario, it is difficult to estimate optimal edge weights for a given graph. This has been discussed in Ghosh et al. (2008) for the sum of the effective resistances of a graph, and in Xiao and Boyd (2003); Boyd et al. (2006) for $\chi_1[\Lambda(\lambda)]$. Similarly to Ghosh et al. (2008), we propose an SDP to minimize the corresponding communication rate, which relies on a slightly technical convexity lemma (see Appendix F.1). However, $\lambda \to \chi_2[\Lambda(\lambda)]$ is not convex. Indeed, the supremum is taken over a set of edge *dynamically* defined through λ, $\mathcal{E}(\lambda) \triangleq \{(i,j), \lambda_{ij} > 0\}$ which makes it non-convex: an optimal solution might lead to removing some edges from the graph and consider the maximum effective resistance of this sub-graph. This forces us to slightly relax this condition by taking the supremum over a predefined *fixed* set of edges $\mathcal{E}$ which does not depend on λ, and we introduce for any $\lambda \in \mathbb{R}_+^{\mathcal{E}}$:

$$\chi_2^{\mathcal{E}}(\lambda) \triangleq \frac{1}{2}\max_{(i,j)\in\mathcal{E}} (e_i - e_j)^\top \Lambda^+(\lambda)(e_i - e_j)\,.$$

Indeed, we have the following Lemma (proved in Appendix F.1), which we could not find in the literature and is slightly technical:

Proposition 6 (log**-convexity lemma)** *Fix $\|u\| = 1$, $u \perp \mathbf{1}$ and let $\lambda = (\lambda_{ij}) \in \mathbb{R}_+^{n^2}$ such that $\Lambda(\lambda) = \sum_{1\le i,j\le n}\lambda_{ij}(e_i - e_j)(e_i - e_j)^\mathsf{T}$, and let:*

$$\psi(\lambda, u) = u^\mathsf{T}\Lambda(\lambda)^+u\,.$$

Then, $\lambda \to \psi(\lambda, u)$ is log-convex on $\mathbb{R}_+^{n^2}$. Furthermore, $\lambda \to \psi(\lambda, u)$ is strictly convex on the convex set $\{\lambda, \chi_1[\Lambda(\lambda)] < \infty\}$ of the connected graphs.

Thus, this function is convex, and we note that by definition $\chi_2^{\mathcal{E}}(\lambda) \ge \chi_2[\Lambda(\lambda)]$. We now propose several structural constraints on the weights of a Laplacian:

($\mathcal{C}_1$) **Global bandwidth:** $\sum_{(ij)\in\mathcal{E}}\lambda_{ij} \le \lambda$, where $\lambda > 0$ is a total bandwidth constraint,

($\mathcal{C}_2$) **Local connectivity:** $\sum_{j,(ij)\in\mathcal{E}} \lambda_{ij} \leq \lambda_i$, where $\lambda_i > 0$ is a local constraint on node connectivity,

($\mathcal{C}_3$) **Local bandwidth:** $\sum_{j,(ij)\in\mathcal{E}} \lambda_{ji} + \lambda_{ij} \leq \lambda_i$, where $\lambda_i > 0$ is a local constraint on edge connectivity to node i.

In the following, we will generically call $\mathcal{C} \in \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3\}$ one of these linear constraints. Getting back to our initial problem, we want to consider:

$$\begin{aligned} \text{minimize} \quad & \operatorname{Tr}\Lambda(\lambda)\sqrt{2\chi_1[\Lambda(\lambda)]\chi_2[\Lambda(\lambda)]} \\ \text{subject to} \quad & \lambda \in \mathcal{C} \cap \mathbb{R}_+^{\mathcal{E}} \end{aligned} \tag{5}$$

which allows connected subgraphs of $\mathcal{E}$ to be considered. However, in order to frame this minimization problem as a SDP, we first need to relax it, see Appendix F.2 for a complete discussion. This leads to the following proposition, proved in Appendix F.3:

Proposition 7 (Minimizing communications as a SDP) *Given a fixed topology $\mathcal{E}$, the problem:*

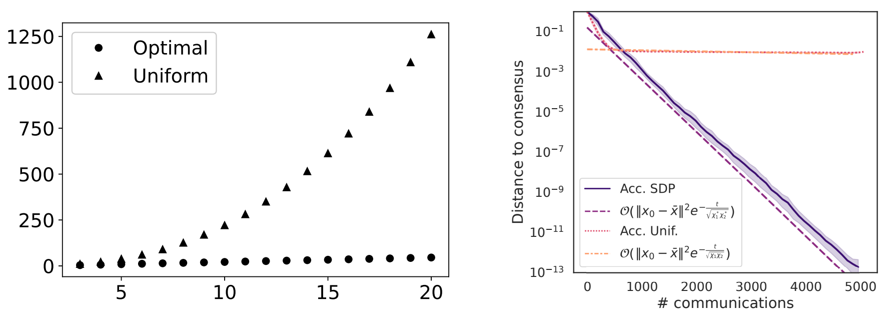
$$\begin{aligned} \textit{minimize} \quad & \sqrt{2\chi_1[\Lambda(\lambda)]\chi_2^{\mathcal{E}}(\lambda)} \\ \textit{subject to} \quad & \lambda \in \mathcal{C} \cap \mathbb{R}_+^{\mathcal{E}} \end{aligned} \tag{6}$$

is equivalent to the following SDP:

$$\begin{aligned} \textit{minimize} \quad & t_1 + t_2 \\ \textit{subject to} \quad & \lambda \in u\mathcal{C}, u \geq 1, \lambda_{ij} \geq 0 \; \forall\, (i,j) \in \mathcal{E} \\ & \Lambda = \textstyle\sum_{ij} \lambda_{ij}(e_i - e_j)(e_i - e_j)^\mathsf{T} \\ & \begin{pmatrix} \Lambda & u(e_i - e_j) \\ u(e_i - e_j)^\mathsf{T} & t_1 \end{pmatrix} \succcurlyeq 0 \quad \forall\, (i,j) \in \mathcal{E} \\ & \begin{pmatrix} \Lambda & \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T} \\ \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T} & t_2\mathbf{I} \end{pmatrix} \succcurlyeq 0 \end{aligned}$$

4.1 Case of the Barbell Graph

In this section, following Ghosh et al. (2008), we make an in-depth study of the communication rates given by our SDP on the Barbell graph $K_n - K_n$ on $2n$ nodes (see Appendix F.5 for results on other graphs found using CVXPY (Diamond and Boyd, 2016)). First, we propose an analytical solution to problem (6) under the global bandwidth constraint $\mathcal{C}_1$. In this setting, we show that the optimal communication strategy can be unboundedly better than using uniform communication rates on the edges. Second, we illustrate the advantage of using these weights in practice by running the accelerated gossip procedure using both gossip matrices. Finally, we illustrate the Braess paradox on the Barbell graph by showing that removing some edges leads to better convergence rates.



(a) (b)

Figure 1: (a) Optimal value of $\sqrt{2\chi_1\chi_2}$ for problem (6) under constraint $\mathcal{C}_1$ compared to the one obtained with uniform weights $1/|\mathcal{E}|$, as a function of n for the barbell graph. (b) Convergence rate for the accelerated gossip procedure on a barbell graph $K_{50}-K_{50}$ using both the edge weights given by our SDP and the uniform ones. For reference, we plot the theoretical rates, denoting by χ_1^*, χ_2^* the values obtained using the SDP weights, and by χ_1, χ_2 the ones obtained with $1/|\mathcal{E}|$. Averaged over 20 runs.

4.1.1 Analytical study

Using the same symmetry argument than Ghosh et al. (2008), we know the problem actually depends on three variables a, b, c as illustrated on Fig. 2. We then have the following Lemma, proved in Appendix F.4, which shows that using uniform weights is sub-optimal on the barbell graph.

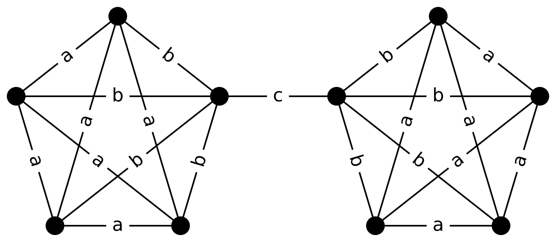


Figure 2: The barbell graph K_5-K_5.

Lemma 8 (Barbell graph) *For K_n-K_n with $n\geq 3$, under the constraint $\sum_{(i,j)\in\mathcal{E}}\lambda_{ij}\leq 1$, the optimal parameters for problem (6) are given by: $c^* = \frac{4-n+\sqrt{2n(n-1)}}{2(8+n)} \underset{n\to\infty}{\sim} \frac{\sqrt{2}-1}{2}$, $b^* = \frac{c^*}{n-2}\left(\sqrt{\frac{2(n-1)}{n}}-\frac{2}{n}\right) \underset{n\to\infty}{\sim} \frac{\sqrt{2}-1}{n\sqrt{2}}$, $a^* = \frac{1-c^*-2b^*(n-1)}{(n-1)(n-2)} \underset{n\to\infty}{\sim} \frac{\sqrt{2}-1}{2n^2}$. In this case, $\sqrt{2\chi_1\chi_2} \underset{n\to\infty}{\simeq} 2.4n$. Using uniform weights $a=b=c=\frac{1}{|\mathcal{E}|}$ leads to $\sqrt{2\chi_1\chi_2} \underset{n\to\infty}{\sim} \frac{n^{5/2}}{\sqrt{2}}$.*

Thus, as reported in Tab. 2, running the accelerated gossip procedures on the Barbell graph using uniform weights is $\mathcal{O}(n^{3/2})$ worse than with weights solution to the relaxed problem (6). This is illustrated in Fig. 1 (a) and confirmed by Fig. 1 (b), which displays the impact of choosing sub-optimal gossip matrices for the convergence rates of gossip-based methods in practice.

4.1.2 Effect of the relaxation

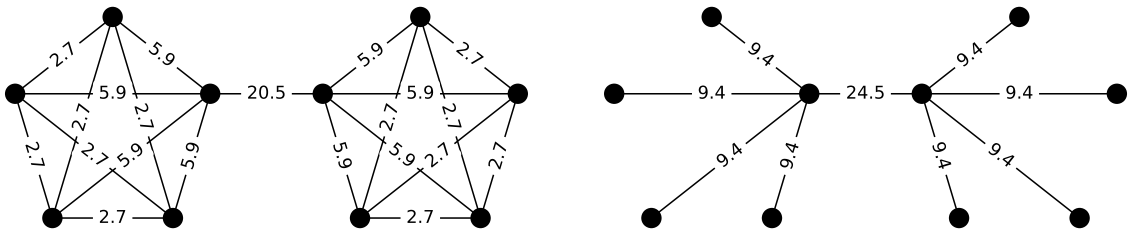


Figure 3: (a) Weights found by our SDP for the Barbell graph $K_5 - K_5$, leading to $\sqrt{\chi_1\chi_2} \simeq 12.9$. (b) Optimal weights for the subgraph where edges a were removed, leading to a value of $\sqrt{\chi_1\chi_2} \simeq 10.2$. For the sake of readability, all values were multiplied by a factor of 100.

As discussed in Rem.19, relaxing our problem from Eq. (5) to Eq. (6) prohibits the removal of edges from the given graph. However, as in traffic networks where doing so can sometimes speed up the overall traffic flow (David and Jon, 2010), considering sub-graphs might lead to better values of $\sqrt{\chi_1\chi_2}$. As it happens, this is actually the case on the Barbell graph. As shown in Fig. 3, removing edges with weight a leads to a better communication complexity. Thus, the relaxation Eq. (6) cannot find the optimal solution of Eq. (5) in this example. However, it is still orders of magnitude better than putting uniform weights on the edges. In fact, a solution of problem Eq. (6) will always lead to faster rates than using any heuristics putting non 0 weight on the edges of the graph. For more numerical results using our SDP solvers, see Appendix F.5.

5. Practical implementation

5.1 Algorithm

To study the trajectories of $X_t \triangleq (x_t, \tilde{x}_t, \tilde{y}_t), Y_t \triangleq (y_t, z_t, \tilde{z}_t)$, we use the following, equivalent to Eq. (3):

$$\begin{aligned} dX_t &= a_1(X_t, Y_t)dt + b_1(X_t)d\mathbf{N}(t) \\ dY_t &= a_2(X_t, Y_t)dt + \sum_{(i,j)\in\mathcal{E}} b_2^{ij}(Y_t)dM_{ij}(t)\,, \end{aligned} \tag{7}$$

where $a_1, a_2, b_1 = (b_1^i)_i, (b_2^{ij})_{ij}$ are smooth functions. We now describe the algorithm used to implement the dynamics of Eq. (3). Let us write $T_1^{(i)} < T_2^{(i)} < ... < T_k^{(i)} < ...$ the time of the k-th event on the i-th node, which corresponds to the spike times of the Poisson processes driven by M_{ij} and $\mathbf{N}$, which is either an edge activation, either a gradient update. We remind that the spiking times of a specific event correspond to random variables with independent exponential increments and can thus be generated at the beginning of our simulation. They can also be generated on the fly and locally to stress the locality and asynchronicity of our algorithm. We write $X_t = (X_t^{(i)})_i$ and $Y_t = (Y_t^{(i)})_i$, then on the i-th node, at the k-th iteration, we integrate on $[T_k^{(i)}; T_{k+1}^{(i)}]$ the ODE

$$\begin{aligned} dX_t &= a_1(X_t, Y_t)dt \\ dY_t &= a_2(X_t, Y_t)dt \end{aligned}$$

to define the values right before the spike. One can easily find the matrix $\mathcal{A} \in \mathbb{R}^{6\times 6}$ (defined formally in Appendix H.2) such that:

$$\begin{pmatrix} X^{(i)}_{T^{(i)-}_{k+1}} \\ Y^{(i)}_{T^{(i)-}_{k+1}} \end{pmatrix} = \exp\left((T^{(i)}_{k+1} - T^{(i)}_k)\mathcal{A}\right) \begin{pmatrix} X^{(i)}_{T^{(i)}_k} \\ Y^{(i)}_{T^{(i)}_k} \end{pmatrix} . \tag{8}$$

Next, if one has a gradient update, then:

$$X^{(i)}_{T^{(i)}_{k+1}} = X^{(i)}_{T^{(i)-}_{k+1}} + b_1\left(X^{(i)}_{T^{(i)-}_{k+1}}\right) .$$

Otherwise, if the edge (i, j) or (j, i) is activated, a communication bridge is created between both nodes i and j. In this case, the local update on i writes:

$$Y^{(i)}_{T^{(i)}_{k+1}} = Y^{(i)}_{T^{(i)-}_{k+1}} + b_2\left(Y^{(i)}_{T^{(i)-}_{k+1}}, Y^{(j)}_{T^{(i)-}_{k+1}}\right) .$$

Note that, even if this event takes place along an edge (i, j), we can write it separately for nodes i and j by making sure they both have the events $T^{(i)}_{k_i} = T^{(j)}_{k_j}$, for some $k_i, k_j \in \mathbb{N}$ corresponding to this communication. As advocated, all those operations are local, and we summarize in the Alg. 1 the algorithmic block corresponding to our implementation. See Appendix H for more details.

5.2 Numerical results

In this section, we study the behavior of our method in a standard experimental setting (*e.g.*, seeKovalev et al. (2021a); Even et al. (2021a)). In order to compare to methods using the gradient of the Fenchel conjugate (Even et al., 2021a) in our experiments, we restrict ourselves to a situation where it is easily computable. Thus, we perform the empirical risk minimization for the decentralized linear regression task given by:

$$f_i(x) = \frac{1}{m}\sum_{j=1}^{m} \|a_{ij}^\top x - c_{ij}\|^2, \tag{9}$$

where $a_{ij} \in \mathbb{R}^d$, and $c_{ij} \in \mathbb{R}$ correspond to m local data points stored at node i. We follow a protocol similar to Kovalev et al. (2021a): we generate n independent synthetic datasets with the `make_regression` functions of scikit-learn (Pedregosa et al., 2011), each worker storing $m = 100$ data points. We recall that the metrics of interest are the total number of local gradient steps and the total number of individual messages exchanged *(*i.e.*, number of edges that fired)* to reach an ϵ-precision. We systematically used the proposed hyper-parameters of each reference paper for our implementation without any specific fine-tuning.

Algorithm 1: This algorithm block describes our implementation on each local machine. The *ODE* routine is described by Eq. 8.

```
   Input: On each machine $i \in \{1, ..., n\}$, gradient oracle $\nabla f_i$, parameters
          $\mu, L, \chi_1, t_{\max}$.
 1 Initialize on each machine $i \in \{1, ..., n\}$:
 2     Set $X^{(i)}, Y^{(i)}, T^{(i)}$ to 0 and set $\mathcal{A}$ via Eq. (90);
 3 Synchronize the clocks of all machines ;
 4 In parallel on workers $i \in \{1, ..., n\}$, while $t < t_{\max}$, continuously do:
 5     $t \leftarrow clock()$ ;
 6     if there is an event at time $t$ then
 7         $(X^{(i)}, Y^{(i)}) \leftarrow ODE(\mathcal{A}, t - T^{(i)}, X^{(i)}, Y^{(i)})$;
 8         if the event is to take a gradient step then
 9             $X^{(i)} \leftarrow X^{(i)} + b_1(X^{(i)})$ ;
10         else if the event is to communicate with $j$ then
11             $Y^{(i)} \leftarrow Y^{(i)} + b_2(Y^{(i)}, Y^{(j)})$ ;          // Happens at $j$ simultaneously.
12         $T^{(i)} \leftarrow t$ ;
13 return $(x^{(i)}_{t_{\max}})_{1 \le i \le n}$.
```

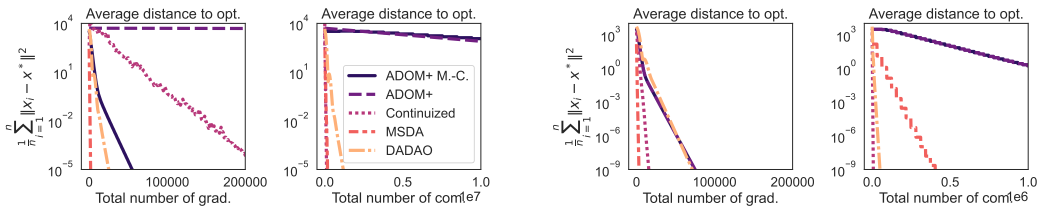


Figure 4: Comparison between ADOM+ (Kovalev et al., 2021a), the continuized framework (Even et al., 2021a), MSDA (Scaman et al., 2017) and DADAO, using the same data for the linear regression task, and the same graphs *(from left to right: line with $n = 150$, complete with $n = 250$)*.

Comparison between all methods. We fix the Laplacian matrix via the paragraph 4 to compare simultaneously to the continuized framework (Even et al., 2021a), MSDA (Scaman et al., 2017) and ADOM+ (Kovalev et al., 2021a). We compare ourselves to both versions of ADOM+: with and without the Multi-Consensus (M.-C.). We report in Fig. 4 results corresponding to the complete graph with $n = 250$ nodes and the line graph of size $n = 150$. While sharing the same asymptotic rate (see Fig. 5 for experimental confirmation), we note that the Continuized framework (Even et al., 2021a) and MSDA (Scaman et al., 2017) have better absolute constants than DADAO, giving them an advantage both in terms of the number of communication and gradient steps. However, in the continuized framework, the gradient and communication steps being coupled, the number of gradient computations can potentially be orders of magnitude worse than our algorithm, which is reflected by Fig. 4 for

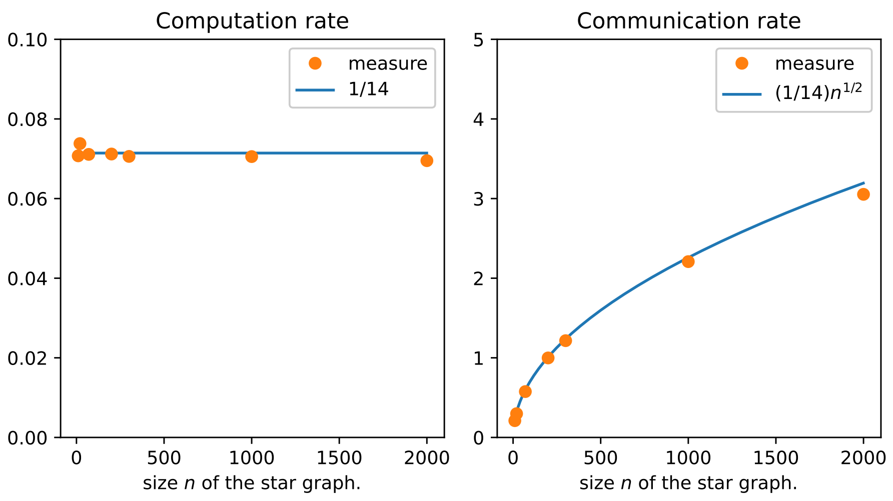


Figure 5: Rate between the slopes in log scale of DADAO and MSDA for star graphs of size $n \in \{10, 20, 70, 200, 300, 1000, 2000\}$.

the line graph. As for MSDA, Tab. 2 showed they do not have the best communication rates on certain classes of graphs, as confirmed to the right in Fig. 4 for MSDA and in Fig. 5. Thanks to its M.-C. procedure, ADOM+ can significantly reduce the number of necessary gradient steps. Yet, consistently with our analysis in Prop. 5, our method is systematically better in all settings in terms of communications.

Further comparison between DADAO and MSDA. To experimentally confirm that our communication complexity is better than accelerated methods using the spectral gap and abstract away the better absolute constants for MSDA, we ran DADAO and MSDA on the task of distributed linear regression for star graphs of size $n \in \{10, 20, 70, 200, 300, 1000, 2000\}$. We considered the evolution of the average distance to the optimal with the number of gradient steps and commmunication steps in log scale for each run, and computed the slope of each line. For each graph size, we report in Fig. 5 the rate between the slope for DADAO and the slope for MSDA. We remark that the rate between the gradient complexities of DADAO and MSDA is indeed a $\mathcal{O}(1)$ (with a constant value $\simeq 1/14$) while MSDA is indeed $\mathcal{O}(\sqrt{n})$ worse than DADAO for communications on the star graph, as stated in Tab. 2.

Discussion. We do not claim the optimality of the absolute constant in DADAO's rate (the $8\sqrt{2}$ term in the exponential of Th. 4), which might be further optimized with a tighter analysis and different values of parameters. However, our focus being on large scale problems, we are concerned about asymptotic rates rather than the effect of the absolute constants. Thus, even-though Fig. 4 may sometimes display a faster convergence for the continuized framework (Even et al., 2021a) and MSDA (Scaman et al., 2017) in terms of *number of steps*, we remind that both (Even et al., 2021a; Scaman et al., 2017) use expensive evaluation of dual gradients whereas we only use primal gradients, and Tab. 2 confirms that DADAO has at least their asymptotic rate. On the star graph, where Tab. 2 showed a strict advantage for DADAO compared to MSDA, Fig. 5 confirms that our convergence rate meets the one of (Scaman et al., 2017) for $n \simeq 200$, and has strictly better rates for larger graphs. Finally, we note that, our algorithm falling into the *randomized* category, further improvements may be theoretically possible. Indeed, Woodworth and Srebro (2016) showed that the lower bound on the number of grad-and-prox oracle accesses for the optimization of composite objectives with deterministic methods is in $\mathcal{O}(n\sqrt{\frac{L}{\mu}}\log(\frac{1}{\epsilon}))$, which is attained by work such as (Scaman et al., 2017). However, for randomized algorithms, the lower bound is in $\mathcal{O}(n+\sqrt{\frac{nL}{\mu}}\log(\frac{1}{\epsilon}))$ and, to the best of our knowledge, no algorithm meeting this bound is known to date in the decentralized setting, and in particular, in the asynchronous one.

In conclusion, while several methods can share similar convergence rates, ours is the only one to perform at least as well as its competitors in every setting for different graph's topology, as predicted by Tab. 1.

6. Conclusion

In this work, we have proposed a novel stochastic algorithm for the decentralized optimization of a sum of smooth and strongly convex functions. We have demonstrated, theoretically and empirically, that this algorithm leads systematically to a substantial acceleration compared to state-of-the-art works. Furthermore, our algorithm is asynchronous, decoupled, primal, and does not rely on an extra inner loop: each of these properties makes it suitable for real applications. We also proposed a novel approach for minimizing the accelerated communication rate of asynchronous gossip algorithms. In future work, we would like to explore the robustness of such algorithms to more challenging variabilities occurring in real-life applications such as time-varying networks and to extend our work to less regular functions and stochastic analysis.

Acknowledgements

This work was supported by Project ANR-21-CE23-0030 ADONIS, PEPR IA on grant SHARP ANR-23-PEIA-0008, and EMERG-ADONIS from Alliance SU. This work was granted access to the HPC/AI resources of IDRIS under the allocation AD011013743 made by GENCI.

References

Ludwig Arnold. Stochastic differential equations. *New York*, 1974.

Necdet Serhat Aybat and Mert Gürbüzbalaban. Decentralized computation of effective resistances and acceleration of consensus algorithms. In *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 538–542, 2017. doi: 10.1109/GlobalSIP.2017.8308701.

Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Decoupled greedy learning of CNNs. 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 736–745. PMLR, 13–18 Jul 2020.

Eugene Belilovsky, Louis Leconte, Lucas Caccia, Michael Eickenberg, and Edouard Oyallon. Decoupled greedy learning of cnns for synchronous and asynchronous distributed learning. *arXiv preprint arXiv:2106.06401*, 2021.

S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6):2508–2530, 2006. doi: 10.1109/TIT.2006.874516.

Stephen Boyd, Persi Diaconis, and Lin Xiao. Fastest mixing markov chain on a graph. *SIAM Review*, 46(4):667–689, 2004.

Bugra Can, Saeed Soori, Necdet Serhat Aybat, Maryam Mehri Dehnavi, and Mert Gürbüzbalaban. Randomized gossiping with effective resistance weights: Performance guarantees and applications. *IEEE Transactions on Control of Network Systems*, 9(2):524–536, 2022.

Ashok K Chandra, Prabhakar Raghavan, Walter L Ruzzo, Roman Smolensky, and Prasoon Tiwari. The electrical resistance of a graph captures its commute and cover times. *computational complexity*, 6(4):312–340, 1996.

Yat-Tin Chow, Wei Shi, Tianyu Wu, and Wotao Yin. Expander graph and communication-efficient decentralized optimization. In *2016 50th Asilomar Conference on Signals, Systems and Computers*, pages 1715–1720. IEEE, 2016.

Laurent Condat, Daichi Kitahara, Andrés Contreras, and Akira Hirabayashi. Proximal splitting algorithms for convex optimization: A tour of recent advances, with new twists, 2019.

Easley David and Kleinberg Jon. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, USA, 2010. ISBN 0521195330.

Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

Wendy Ellens, Floske M Spieksma, Piet Van Mieghem, Almerima Jamakovic, and Robert E Kooij. Effective graph resistance. *Linear algebra and its applications*, 435(10):2491–2506, 2011.

Mathieu Even, Hadrien Hendrikx, and Laurent Massoulié. Asynchrony and acceleration in gossip algorithms, 2020.

Mathieu Even, Raphaël Berthier, Francis Bach, Nicolas Flammarion, Hadrien Hendrikx, Pierre Gaillard, Laurent Massoulié, and Adrien Taylor. A continuized view on nesterov acceleration for stochastic gradient descent and randomized gossip. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021a.

Mathieu Even, Hadrien Hendrikx, and Laurent Massoulie. Decentralized optimization with heterogeneous delays: a continuous-time approach. *arXiv preprint arXiv:2106.03585*, 2021b.

Marguerite Frank. The braess paradox. *Mathematical Programming*, 20(1):283–302, 1981.

Arpita Ghosh, Stephen Boyd, and Amin Saberi. Minimizing effective resistance of a graph. *SIAM review*, 50(1):37–66, 2008.

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.

Hadrien Hendrikx. A principled framework for the design and analysis of token algorithms, 2022.

Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. An accelerated decentralized stochastic proximal algorithm for finite sums. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019a.

Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. Accelerated decentralized optimization with local updates for smooth and strongly convex objectives. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 897–906. PMLR, 2019b.

Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. Dual-free stochastic decentralized optimization with variance reduction. *Advances in neural information processing systems*, 33:19455–19466, 2020.

Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. An optimal algorithm for decentralized finite-sum optimization. *SIAM Journal on Optimization*, 31(4):2753–2783, 2021. doi: 10.1137/20M134842X.

Douglas J Klein. Resistance-distance sum rules. *Croatica chemica acta*, 75(2):633–649, 2002.

Douglas J Klein and Milan Randić. Resistance distance. *Journal of mathematical chemistry*, 12(1):81–95, 1993.

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.

Anastasiia Koloskova, Tao Lin, and Sebastian U Stich. An improved analysis of gradient tracking for decentralized machine learning. *Advances in Neural Information Processing Systems*, 34:11422–11435, 2021.

Dmitry Kovalev, Adil Salim, and Peter Richtarik. Optimal and practical algorithms for smooth and strongly convex decentralized optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18342–18352. Curran Associates, Inc., 2020.

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 A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021a.

Dmitry Kovalev, Alexander Gasnikov, and Peter Richtárik. Accelerated primal-dual gradient method for smooth and convex-concave saddle-point problems with bilinear coupling. *arXiv preprint arXiv:2112.15199*, 2021b.

Dmitry Kovalev, Egor Shulgin, Peter Richtárik, Alexander V Rogozin, and Alexander Gasnikov. Adom: accelerated decentralized optimization method for time-varying networks. In *International Conference on Machine Learning*, pages 5784–5793. PMLR, 2021c.

Günter Last and Mathew Penrose. *Lectures on the Poisson process*, volume 7. Cambridge University Press, 2017.

Jonas Latz. Analysis of stochastic gradient descent in continuous time. *Statistics and Computing*, 31(4):1–25, 2021.

Batiste Le Bars, Aurélien Bellet, Marc Tommasi, Erick Lavoie, and Anne-Marie Kermarrec. Refined convergence and topology learning for decentralized sgd with heterogeneous data. 2022.

Rémi Leblond, Fabian Pedregosa, and Simon Lacoste-Julien. Improved asynchronous parallel optimization analysis for stochastic incremental methods. *arXiv preprint arXiv:1801.03749*, 2018.

Huan Li and Zhouchen Lin. Accelerated gradient tracking over time-varying graphs for decentralized optimization. *arXiv preprint arXiv:2104.02596*, 2021.

Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. A universal catalyst for first-order optimization. *Advances in neural information processing systems*, 28, 2015.

Alexander Lubotzky. Expander graphs in pure and applied mathematics. *Bulletin of the American Mathematical Society*, 49(1):113–162, 2012.

Konstantin Mishchenko, Grigory Malinovsky, Sebastian Stich, and Peter Richtárik. Proxskip: Yes! local gradient steps provably lead to communication acceleration! finally! *arXiv preprint arXiv:2202.09357*, 2022.

Adel Nabli and Edouard Oyallon. Dadao: Decoupled accelerated decentralized asynchronous optimization. *ICML*, 2023.

Adel Nabli, Eugene Belilovsky, and Edouard Oyallon. Acid: Accelerating asynchronous communication in decentralized deep learning. *Advances in Neural Information Processing Systems*, 36:47451–47474, 2023.

Adel Nabli, Louis Fournier, Pierre ERBACHER, Louis Serrano, Eugene Belilovsky, and Edouard Oyallon. ACCO: Accumulate while you communicate, hiding communications in distributed LLM training, 2025.

Angelia Nedic, Alex Olshevsky, and Wei Shi. Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM Journal on Optimization*, 27(4):2597–2633, 2017.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Adil Salim, Laurent Condat, Dmitry Kovalev, and Peter Richtárik. An optimal algorithm for strongly convex minimization under affine constraints. In *International Conference on Artificial Intelligence and Statistics*, pages 4482–4498. PMLR, 2022.

Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal algorithms for smooth and strongly convex distributed optimization in networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3027–3036. PMLR, 06–11 Aug 2017.

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.

Zhuoqing Song, Lei Shi, Shi Pu, and Ming Yan. Optimal gradient tracking for decentralized optimization. *arXiv preprint arXiv:2110.05282*, 2021.

Sebastian U. Stich and Sai Praneeth Karimireddy. The error-feedback framework: Sgd with delayed gradients. *Journal of Machine Learning Research*, 21(237):1–36, 2020.

Zhenheng Tang, Shaohuai Shi, and Xiaowen Chu. Communication-efficient decentralized learning with sparsification and adaptive peer selection. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, pages 1207–1208. IEEE, 2020.

Thijs Vogels, Hadrien Hendrikx, and Martin Jaggi. Beyond spectral gap: The role of the topology in decentralized learning. *arXiv preprint arXiv:2206.03093*, 2022.

Vaya Sapobi Samui Vos. Methods for determining the effective resistance. *Mestrado, Mathematisch Instituut Universiteit Leiden*, 2016.

Blake E Woodworth and Nati Srebro. Tight complexity bounds for optimizing composite objectives. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Lin Xiao and S. Boyd. Fast linear iterations for distributed averaging. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, volume 5, pages 4997–5002 Vol.5, 2003.

Lin Xiao, Stephen Boyd, and Seung-Jean Kim. Distributed average consensus with least-mean-square deviation. *Journal of Parallel and Distributed Computing*, 67(1):33–46, 2007. ISSN 0743-7315.

Jiaqi Zhang and Keyou You. Fully asynchronous distributed optimization with linear convergence in directed networks, 2019.

Appendix

Table of Contents

A Communication bounds **24**

B Saddle Point Reformulation **26**

C Proof of the main Theorem (Th. 4) **26**
C.1 Proof of the Lemma 12 29

D Computation of graph relevant quantities for standard graphs **34**

E Comparison of complexities with related work **35**

F Optimizing the communication rates with a SDP **38**
F.1 Convexity lemmas 38
F.2 Problem relaxation 40
F.3 SDP formulation 41
F.4 Optimal weights for the Barbell Graph 42
F.5 Results from our SDP solver 45

G Physical interpretation of the sufficient condition **45**

H Practical Implementation **46**
H.1 Simulating the Poisson Point Processes 46
H.2 Pseudo Code 47

Appendix A. Communication bounds

The following properties will be used all along the proofs of the Lemmas and Theorems and are related to the communication of our nodes.

Lemma 9 *Under the assumptions of Theorem 4, if $z_0, \tilde{z}_0 \in span(\pi)$, then $z_t, \tilde{z}_t \in span(\pi)$ almost surely.*

Proof It's clear that for any i, j, we get:

$$\pi(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^\mathsf{T} = (\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^\mathsf{T}\,.$$

Thus, the variations of $(z_t, \tilde{z}_t)$ belong to span(π), and so is the trajectory. ■

Lemma 10 (Effective resistance contraction) *For $(i,j)\in\mathcal{E}$ and any $x\in\mathbb{R}^d$, we have:*

$$\|(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}x\|^2_{\mathbf{\Lambda}^+}\leq \chi_2\|(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}x\|^2\,.$$

Proof Indeed, we note that:

$$\begin{aligned}\|(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}x\|^2_{\mathbf{\Lambda}^+} &= x^\mathsf{T}(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}\mathbf{\Lambda}^+(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}x &(10)\\ &\leq 2\chi_2 x^\mathsf{T}(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}x &(11)\\ &= \chi_2\|(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}x\|^2 &(12)\end{aligned}$$

■

Lemma 11 (Bound on the resistance)
For $(i,j)\in\mathcal{E}$, $(e_i-e_j)^\mathsf{T}\Lambda^+(e_i-e_j)\leq\frac{1}{\lambda_{ij}+\lambda_{ji}}$.

Proof For a graph such that $|\bar{\mathcal{E}}|=1$, the inequality is trivial. Now, we assume that there are other edges than (i,j) or (j,i). Thus, we have since $\lambda_{ij}\geq 0$:

$$\Lambda \succcurlyeq (\lambda_{ij}+\lambda_{ji})(e_i-e_j)(e_i-e_j)^\mathsf{T}\,.$$

In this case: $\big((\lambda_{ij}+\lambda_{ji})(e_i-e_j)(e_i-e_j)^\mathsf{T}\big)^+\succcurlyeq\Lambda^+$, yet:

$$\Big((\lambda_{ij}+\lambda_{ji})(e_i-e_j)(e_i-e_j)^\mathsf{T}\Big)^+=\frac{1}{4(\lambda_{ij}+\lambda_{ji})}(e_i-e_j)(e_i-e_j)^\mathsf{T}\,,$$

and this implies that $\frac{1}{\lambda_{ij}+\lambda_{ji}}\geq(e_i-e_j)^\mathsf{T}\Lambda^+(e_i-e_j)$. ■

Proof [Proof of Lemma 2] First, we note that Λ is symmetric and has a non-negative spectrum, as:

$$x^\mathsf{T}\Lambda x=\sum_{(ij)\in\mathcal{E}}\lambda_{ij}\|x_i-x_j\|^2\,.$$

From this, we also clearly see that $\chi_1=+\infty$ iff the graph is disconnected. Next, assuming that the graph is connected, 0 is an eigenvalue of Λ with multiplicity 1 and by definition of χ_1, we have $\chi_1\geq\chi_2$. As we also have the following by linearity of the Trace:

$$\begin{aligned}\sum_{(i,j)\in\mathcal{E}}\lambda_{ij}(e_i-e_j)^\mathsf{T}\Lambda^+(e_i-e_j) &= \sum_{(i,j)\in\mathcal{E}}\lambda_{ij}\operatorname{Tr}\Big[\Lambda^+(e_i-e_j)(e_i-e_j)^\mathsf{T}\Big]\\ &=\operatorname{Tr}\left(\Lambda^+\sum_{(i,j)\in\mathcal{E}}\lambda_{ij}(e_i-e_j)(e_i-e_j)^\mathsf{T}\right)\\ &=\operatorname{Tr}(\Lambda^+\Lambda)\\ &=n-1\,, &\text{(Resistance)}\end{aligned}$$

we can write:

$$n - 1 \leq 2\chi_2 \sum_{(i,j)\in\mathcal{E}} \lambda_{ij} = \chi_2 \mathrm{Tr}\,\Lambda$$

and get $\frac{n-1}{\mathrm{Tr}\,\Lambda} \leq \chi_2$. Finally, note that: $\mathrm{Tr}\,(\Lambda) = 2\sum_{(i,j)\in\mathcal{E}} \lambda_{ij} \leq 2|\bar{\mathcal{E}}| \sup_{(i,j)\in\mathcal{E}}(\lambda_{ij} + \lambda_{ji})$ and $\mathrm{Tr}\,(\Lambda) \leq (n-1)\|\Lambda\|$, so that, using Lemma 11:

$$\sqrt{\chi_2}\mathrm{Tr}(\Lambda) \leq \frac{1}{\sqrt{2\inf_{(i,j)\in\mathcal{E}}(\lambda_{ij} + \lambda_{ji})}}\sqrt{\mathrm{Tr}\,\Lambda}\sqrt{\mathrm{Tr}\,\Lambda} \tag{13}$$

$$\leq \sqrt{\frac{\mathrm{Tr}\,\Lambda}{2\inf_{(i,j)\in\mathcal{E}}(\lambda_{ij} + \lambda_{ji})}}\sqrt{2|\bar{\mathcal{E}}| \sup_{(i,j)\in\mathcal{E}}(\lambda_{ij} + \lambda_{ji})} \tag{14}$$

$$\leq \sqrt{\kappa}\sqrt{|\bar{\mathcal{E}}|}\sqrt{(n-1)\|\Lambda\|} \tag{15}$$

■

Appendix B. Saddle Point Reformulation

With $0 < \nu < \mu$ and introducing an extra dual variable $\hat{x}$, we get:

$$\begin{aligned}
\inf_{x\in\mathbb{R}^d} \sum_{i=1}^n f_i(x) &= \inf_{\substack{x,\hat{x}\in\mathbb{R}^{n\times d}\\ x=\hat{x},\pi\hat{x}=0}} \sum_{i=1}^n f_i(x_i) - \frac{\nu}{2}\|x\|^2 + \frac{\nu}{2}\|\hat{x}\|^2 \\
&= \inf_{x,\hat{x}\in\mathbb{R}^{n\times d}} \sup_{y,z\in\mathbb{R}^{n\times d}} \sum_{i=1}^n f_i(x_i) - \frac{\nu}{2}\|x\|^2 + \frac{\nu}{2}\|\hat{x}\|^2 + \langle y, \hat{x} - x\rangle + \langle z, \pi\hat{x}\rangle \\
&= \inf_{x\in\mathbb{R}^{n\times d}} \sup_{y,z\in\mathbb{R}^{n\times d}} \inf_{\hat{x}\in\mathbb{R}^{n\times d}} \sum_{i=1}^n f_i(x_i) - \frac{\nu}{2}\|x\|^2 + \frac{\nu}{2}\|\hat{x}\|^2 + \langle y, \hat{x} - x\rangle + \langle z, \pi\hat{x}\rangle \\
&= \inf_{x\in\mathbb{R}^{n\times d}} \sup_{y,z\in\mathbb{R}^{n\times d}} \sum_{i=1}^n f_i(x_i) - \frac{\nu}{2}\|x\|^2 - \langle x, y\rangle - \frac{1}{2\nu}\|\pi z + y\|^2 .
\end{aligned}$$

Note that no specific assumptions were made on ν which can span any values between 0 and μ.

Appendix C. Proof of the main Theorem (Th. 4)

Basic reminds about the Bregman divergence For a smooth convex function F,

$$d_F(x,y) \triangleq F(x) - F(y) - \langle \nabla F(y), x - y\rangle$$

is its Bregman divergence. Next, we note that for $F(x) = \sum_{i=1}^n f_i(x_i) - \frac{\nu}{2}\|x\|^2$, then F is L-smooth, $\mu - \nu$-strongly convex, and we get:

$$\frac{1}{2L}\|\nabla F(x) - \nabla F(y)\|^2 \leq d_F(x,y) \leq \frac{L}{2}\|x - y\|^2 ,$$

and

$$\frac{\mu-\nu}{2}\|x-y\|^2 \leq d_F(x,y) \leq \frac{1}{2(\mu-\nu)}\|\nabla F(x)-\nabla F(y)\|^2\,,$$

In the following, we will set $\nu = \frac{\mu}{2}$.

Proof [Proof of Theorem 4] We introduce for a positive semi-definite matrix A, $\|x\|_A^2 \triangleq x^\mathsf{T}Ax$. For our proof, we rely on the notation of Eq. 7, and we introduce $X \triangleq (x,\tilde{x},\tilde{y}), Y \triangleq (y,z,\tilde{z})$ and the following Lyapunov potential:

$$\begin{aligned}\Phi(t,X,Y) \triangleq\ & A_t d_F(x,x^*) + \tilde{A}_t\|\tilde{x}-x^*\|^2 + B_t\|y-y^*\|^2 + \tilde{B}_t\|\tilde{y}-y^*\|^2 \\ & + C_t\|z+y-z^*-y^*\|^2 + \tilde{C}_t\|\tilde{z}-z^*\|^2_{\mathbf{\Lambda}^+}\,,\end{aligned}$$

where $A_t,\tilde{A}_t,B_t,\tilde{B}_t,C_t,\tilde{C}_t$ are non-negative functions to be defined. We will use this potential to control the trajectories.

Because Φ is smooth, the SDE is a smooth trajectory, we get via Ito's formula (Last and Penrose, 2017) applied to the semi-martingale (X_t,Y_t) on any intervals $[0,T]$:

$$\begin{aligned}\Phi(T,X_T,Y_T) = \Phi(0,X_0,Y_0) &+ \int_0^T \langle\nabla\Phi(t,X_t,Y_t), \begin{pmatrix}1\\ a_1(X_t,Y_t)\\ a_2(X_t,Y_t)\end{pmatrix}\rangle dt \\ &+ \sum_{i=1}^n\int_0^T \big(\Phi(t,X_t+b_1^i(X_t),Y_t)-\Phi(t,X_t,Y_t)\big)dt \\ &+ \sum_{(i,j)\in\mathcal{E}}\int_0^T \big(\Phi(t,X_t,Y_t+b_2^{ij}(Y_t))-\Phi(t,X_t,Y_t)\big)\lambda_{ij}dt \\ &+ \Theta_T\,,\end{aligned}$$

where the following quantity is a Martingale:

$$\begin{aligned}\Theta_u \triangleq & \sum_{i=1}^n\int_0^u \big(\Phi(t,X_{t^-},Y_{t^-}+b_1^i(X_{t^-}))-\Phi(u,X_{t^-},Y_{t^-})\big)(dN_i(t)-dt) \\ &+ \sum_{(i,j)\in\mathcal{E}}\int_0^u \big(\Phi(t,X_{t^-}+b_2^{ij}(X_{t^-}),Y_{t^-})-\Phi(t,X_{t^-},Y_{t^-})\big)(dM_{ij}(t)-\lambda_{ij}dt)\,.\end{aligned}$$

Taking the expectation, we get that, as the initialization is deterministic:

$$\begin{aligned}\mathbb{E}[\Phi(T,X_T,Y_T)] = \Phi(0,X_0,Y_0) &+ \int_0^T \langle\nabla\Phi(t,X_t,Y_t), \begin{pmatrix}1\\ a_1(X_t,Y_t)\\ a_2(X_t,Y_t)\end{pmatrix}\rangle dt \\ &+ \sum_{i=1}^n\int_0^T \big(\Phi(t,X_t+b_1^i(X_t),Y_t)-\Phi(t,X_t,Y_t)\big)dt \\ &+ \sum_{(i,j)\in\mathcal{E}}\int_0^T \big(\Phi(t,X_t,Y_t+b_2^{ij}(Y_t))-\Phi(t,X_t,Y_t)\big)\lambda_{ij}dt\,,\end{aligned}$$

To show that the integrand term is negative, we will use the following technical Lemma, which is also difficult to prove and whose proof is deferred to Appendix C.1:

Lemma 12 *If:*

$$\begin{array}{llll}
\eta = \frac{1}{8\sqrt{2}}\sqrt{\frac{\nu}{\chi_1\chi_2 L}} & \gamma = \frac{1}{4L} & \delta = \frac{1}{4\sqrt{2}}\sqrt{\frac{\nu}{\chi_1\chi_2 L}} & \alpha = \frac{1}{8\sqrt{2}}\sqrt{\frac{\nu}{\chi_1\chi_2 L}} \\
\beta = \frac{1}{2} & \theta = \frac{1}{2\sqrt{2}}\sqrt{\frac{L}{\chi_1\chi_2\nu}} & \tilde{\tau} = \frac{1}{8} & \tilde{\eta} = \frac{1}{8}\sqrt{\frac{\nu}{L}} \\
\tilde{\gamma} = \frac{1}{4\sqrt{2}\sqrt{\nu L\chi_1\chi_2}} & \tilde{\delta} = 1 & \tilde{\alpha} = \frac{1}{8\sqrt{2}}\sqrt{\frac{\nu}{\chi_1\chi_2 L}} & \tilde{\beta} = \sqrt{2\frac{\chi_1 L}{\chi_2\nu}} \\
\nu = \frac{\mu}{2} & \tau = \frac{1}{8} & &
\end{array}$$

and

$$\begin{array}{lll}
A_t = e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} & \tilde{A}_t = \frac{\mu}{8}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} & \tilde{B}_t = \frac{1}{8L}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} \\
B_t = \frac{1}{16L}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} & C_t = \frac{1}{\mu}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} & \tilde{C}_t = \frac{1}{32\chi_1 L}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} .
\end{array}$$

then:

$$\begin{aligned}
&\langle \nabla\Phi(t, X_t, Y_t), \begin{pmatrix} 1 \\ a_1(X_t, Y_t) \\ a_2(X_t, Y_t) \end{pmatrix} \rangle + \big(\Phi(t, X_t + b_1(X_t), Y_t) - \Phi(t, X_t, Y_t)\big) \\
&\quad + \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}\big(\Phi(t, X_t, Y_t + b_2^{ij}(Y_t)) - \Phi(t, X_t, Y_t)\big) \leq 0 \ \ a.s. \ .
\end{aligned}$$

Now, we remark that if we have $F(x) = f(x) - \frac{\mu}{4}\|x\|^2$, and initialize with $\tilde{x}_0 = x_0$, $y_0 = \tilde{y}_0 = \nabla F(x_0)$ and $z_0 = \tilde{z}_0 = -\pi\nabla F(x_0)$, then, given the linear relation between $A_t, \tilde{A}_t, B_t, \tilde{B}_t, C_t, \tilde{C}_t$, the L smoothness and the fact π is an orthogonal projection, we get:

$$\begin{aligned}
\Phi(0, X_0, Y_0) &\leq d_F(x_0, x^*) + \frac{\mu}{8}\|x_0 - x^*\|^2 + \frac{1}{8L}\|\nabla F(x_0) - \nabla F(x^*)\|^2 \\
&\quad + \frac{1}{16L}\|\nabla F(x_0) - \nabla F(x^*)\|^2 + \frac{1}{\mu}\|(\mathbf{I} - \pi)(\nabla F(x_0) - \nabla F(x^*))\|^2 \\
&\quad + \frac{1}{32}\frac{\chi_1}{L\chi_1}\|\pi(\nabla F(x_0) - \nabla F(x^*))\|^2 \\
&\leq (\frac{L}{2} + \frac{\mu}{8} + \frac{L}{8} + \frac{L}{16} + \frac{L^2}{\mu} + \frac{L}{32})\|x_0 - x^*\|^2
\end{aligned}$$

In particular, as $\frac{\mu}{4}\|x - x^*\|^2 \leq d_F(x, x^*)$ it implies that:

$$\mathbb{E}[\|x_t - x^*\|^2] \leq 4(1 + \frac{L}{\mu} + \frac{L^2}{\mu^2})\|x_0 - x^*\|^2 e^{-\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}}$$

Finally, we note that the expected number of gradients between $[0, T]$ is given by:

$$\mathbb{E}[\sum_{i=1}^{n} N_i(T)] = nT \, ,$$

and similarly, the number of edges activated is given by:

$$\mathbb{E}[\sum_{1\le i,j\le n} M_{ij}(T)] = \sum_{1\le i,j\le n}\int_0^T \lambda_{ij}\,dt = \frac{T}{2}\operatorname{Tr}\Lambda.$$

■

C.1 Proof of the Lemma 12

We first state a couple of inequalities that we will combine to obtain a bound on our Lyapunov function. In all this section, for a given variable x, we denote by x^+ the value of x right after a Poisson update.

Lemma 13 *First:*

$$\begin{aligned}\phi_A &\triangleq A_t(d_F(x^+,x^*)-d_F(x,x^*))+\tilde{A}_t(\|\tilde{x}^+-x^*\|^2-\|\tilde{x}-x^*\|^2)\\ &+\eta A_t\langle \tilde{x}-x,\nabla F(x)-\nabla F(x^*)\rangle+2\tilde{\eta}\tilde{A}_t\langle x-\tilde{x},\tilde{x}-x^*\rangle \qquad (16)\\ &\le \|\nabla F(x)-\tilde{y}\|^2\left(A_t\frac{L\gamma^2}{2}-A_t\gamma+\tilde{A}_t\tilde{\gamma}^2\right)\\ &+A_t\gamma\langle\nabla F(x)-\tilde{y},y^*-\tilde{y}\rangle+2\tilde{\gamma}\tilde{A}_t\langle\tilde{y}-y^*,\tilde{x}-x^*\rangle \qquad (17)\\ &-2\tilde{\gamma}\tilde{A}_t\left(d_F(\tilde{x},x^*)+d_F(x^*,x)-d_F(\tilde{x},x)\right)\\ &-\eta A_t(d_F(\tilde{x},x)+d_F(x,x^*)-d_F(\tilde{x},x^*))-\tilde{A}_t\tilde{\eta}\|\tilde{x}-x^*\|^2+\tilde{A}_t\tilde{\eta}\|x-x^*\|^2\end{aligned}$$

Proof First, we have to use optimality conditions and smoothness, as well as the separability of F:

$$\begin{aligned}d_F(x^+,x^*)-d_F(x,x^*) &= d_F(x^+,x)-\langle x^+-x,\nabla F(x^*)-\nabla F(x)\rangle \qquad (18)\\ &\le \frac{L}{2}\|x^+-x\|^2-\langle x^+-x,\nabla F(x^*)-\nabla F(x)\rangle \qquad (19)\\ &= \frac{L\gamma^2}{2}\|\tilde{y}-\nabla F(x)\|^2-\gamma\|\nabla F(x)-\tilde{y}\|^2\\ &+\gamma\langle\nabla F(x)-\tilde{y},y^*-\tilde{y}\rangle \qquad (20)\end{aligned}$$

Next, we note that, again using optimality conditions:

$$\begin{aligned}\|\tilde{x}^+-x^*\|^2-\|\tilde{x}-x^*\|^2 &= 2\langle\tilde{x}^+-\tilde{x},\tilde{x}-x^*\rangle+\|\tilde{x}^+-\tilde{x}\|^2 \qquad (21)\\ &= -2\tilde{\gamma}\langle\nabla F(x)-\tilde{y},\tilde{x}-x^*\rangle+\tilde{\gamma}^2\|\nabla F(x)-\tilde{y}\|^2 \qquad (22)\\ &= -2\tilde{\gamma}\langle\nabla F(x)-\nabla F(x^*),\tilde{x}-x^*\rangle+2\tilde{\gamma}\langle\tilde{y}-y^*,\tilde{x}-x^*\rangle+\tilde{\gamma}^2\|\nabla F(x)-\tilde{y}\|^2 \qquad (23)\\ &= -2\tilde{\gamma}(d_F(\tilde{x},x^*)+d_F(x^*,x)-d_F(\tilde{x},x))+2\tilde{\gamma}\langle\tilde{y}-y^*,\tilde{x}-x^*\rangle+\tilde{\gamma}^2\|\nabla F(x)-\tilde{y}\|^2 \qquad (24)\end{aligned}$$

Momentum in x associated with the term $d_F(x,x^*)$ gives:

$$\eta\langle\tilde{x}-x,\nabla F(x)-\nabla F(x^*)\rangle = -\eta(d_F(\tilde{x},x)+d_F(x,x^*)-d_F(\tilde{x},x^*)) \qquad (25)$$

and momentum in $\tilde{x}$ associated with $\|\tilde{x}-x^*\|^2$ leads to:

$$2\tilde{\eta}\langle x-\tilde{x},\tilde{x}-x^*\rangle = -2\tilde{\eta}\|\tilde{x}-x^*\|^2 + 2\tilde{\eta}\langle x-x^*,\tilde{x}-x^*\rangle \leq -\tilde{\eta}\|\tilde{x}-x^*\|^2 + \tilde{\eta}\|x-x^*\|^2 \tag{26}$$

■

Lemma 14 *Next, we show that if $\alpha B_t = \frac{\delta}{2}\tilde{B}_t$:*

$$\begin{aligned}\phi_B \triangleq\ & B_t(\|y^+-y^*\|^2 - \|y-y^*\|^2) + \tilde{B}_t(\|\tilde{y}^+-y^*\|^2 - \|\tilde{y}-y^*\|^2)\\ & + 2\alpha B_t\langle y-y^*,\tilde{y}-y\rangle - 2\theta\tilde{B}_t\langle y+z+\nu\tilde{x},\tilde{y}-y^*\rangle + 2\alpha C_t\langle\tilde{y}-y,z+y-y^*-z^*\rangle \qquad (27)\\ \leq & -\frac{\delta}{2}\tilde{B}_t\|\tilde{y}-y^*\|^2 - \frac{\delta}{2}\tilde{B}_t\|y-y^*\|^2 - 2\tilde{\delta}\tilde{B}_t\langle\nabla F(x)-\tilde{y},y^*-\tilde{y}\rangle\\ & + \delta\tilde{B}_t\|\nabla F(x)-\nabla F(x^*)\|^2 + \left((\delta+\tilde{\delta})^2-\delta\right)\tilde{B}_t\|\nabla F(x)-y\|^2\\ & - 2\theta\tilde{B}_t\langle y+z-y^*-z^*,\tilde{y}-y^*\rangle - 2\theta\nu\tilde{B}_t\langle\tilde{x}-x^*,\tilde{y}-y^*\rangle + 2\alpha C_t\langle\tilde{y}-y,z+y-y^*-z^*\rangle \qquad (28)\end{aligned}$$

Proof Using optimality conditions:

$$\begin{aligned}\|\tilde{y}^+-y^*\|^2 - \|\tilde{y}-y^*\|^2 &= 2\langle\tilde{y}-y^*,\tilde{y}^+-\tilde{y}\rangle + \|\tilde{y}^+-\tilde{y}\|^2 \qquad (29)\\ &= 2\delta\langle\nabla F(x)-\tilde{y},\tilde{y}-y^*\rangle + 2\tilde{\delta}\langle\nabla F(x)-\tilde{y},\tilde{y}-y^*\rangle + (\delta+\tilde{\delta})^2\|\nabla F(x)-\tilde{y}\|^2 \qquad (30)\\ &= -2\tilde{\delta}\langle\nabla F(x)-\tilde{y},y^*-\tilde{y}\rangle + \delta\|\nabla F(x)-\nabla F(x^*)\|^2 - \delta\|\tilde{y}-y^*\|^2\\ &\quad + \left((\delta+\tilde{\delta})^2-\delta\right)\|\nabla F(x)-\tilde{y}\|^2 \qquad (31)\end{aligned}$$

The momentum in $\tilde{y}$ associated with the term $\|\tilde{y}-y^*\|^2$ gives,thanks to optimality conditions:

$$-2\theta\tilde{B}_t\langle y+z+\nu\tilde{x},\tilde{y}-y^*\rangle = -2\theta\tilde{B}_t\langle y+z-y^*-z^*,\tilde{y}-y^*\rangle - 2\theta\nu\tilde{B}_t\langle\tilde{x}-x^*,\tilde{y}-y^*\rangle \tag{32}$$

The momentum in y associated with the term $\|y-y^*\|^2$ gives:

$$2\alpha B_t\langle\tilde{y}-y,y-y^*\rangle = -\alpha B_t\|y-y^*\|^2 - \alpha B_t\|\tilde{y}-y\|^2 + \alpha B_t\|\tilde{y}-y^*\|^2 \tag{33}$$

and the one associated with $\|y+z-y^*-z^*\|^2$, thanks to optimality conditions can be written:

$$2\alpha C_t\langle\tilde{y}-y,z+y-y^*-z^*\rangle \tag{34}$$

■

Lemma 15 *Finally, assuming* $\theta \tilde{B}_t = \tilde{\beta}\tilde{C}_t = \alpha C_t$*, letting* $1 \geq \tilde{\tau} > 0$*,* $z_{ij}^+ = z - \beta(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^\mathsf{T}(y+z)$ *and* $\tilde{z}_{ij}^+ = \tilde{z} - \tilde{\beta}(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^\mathsf{T}(y+z)$*:*

$$\begin{aligned}
&\phi_C - 2\theta\tilde{B}_t\langle y+z-y^*-z^*, \tilde{y}-y^*\rangle + 2\alpha C_t\langle \tilde{y}-y, z+y-y^*-z^*\rangle \triangleq \\
&\quad \sum_{ij}\lambda_{ij}C_t\Big(\|y+z_{ij}^+-y^*-z^*\|^2 - \|y+z-y^*-z^*\|^2\Big) \\
&\quad + \sum_{ij}\lambda_{ij}\tilde{C}_t\Big(\|\tilde{z}_{ij}^+-z^*\|^2_{\mathbf{\Lambda}^+} - \|\tilde{z}-z^*\|^2_{\mathbf{\Lambda}^+}\Big) + 2\tilde{\alpha}\tilde{C}_t\langle z-\tilde{z}, \tilde{z}-z^*\rangle_{\mathbf{\Lambda}^+} \qquad (35)\\
&\quad + 2\alpha C_t\langle \tilde{z}+\tilde{y}-z-y, z+y-y^*-z^*\rangle - 2\theta\tilde{B}_t\langle y+z-y^*-z^*, \tilde{y}-y^*\rangle \\
&\quad \leq \tilde{\beta}^2\chi_2\tilde{C}_t\sum_{(i,j)\in\mathcal{E}}\lambda_{ij}\|(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}(y+z)\|^2 \\
&\quad + \beta(\beta-1)C_t\sum_{(i,j)\in\mathcal{E}}\lambda_{ij}\|(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}(y+z)\|^2 \\
&\quad - \alpha C_t\|y+z-y^*-z^*\|^2 + \tilde{\alpha}\chi_1\tilde{C}_t\|z-z^*\|^2 - \tilde{\alpha}\tilde{C}_t\|\tilde{z}-z^*\|^2_{\mathbf{\Lambda}^+} \\
&\quad - \tilde{\tau}\frac{1}{2}\tilde{\beta}\frac{\nu}{L}\tilde{C}_t\|z-z^*\|^2 + \tilde{\tau}\frac{\nu}{L}\frac{2\alpha\theta}{\delta}B_t\|y-y^*\|^2 \qquad (36)
\end{aligned}$$

Proof

Having in mind that $\pi(y^*+z^*)=0$ and $\mathbf{\Lambda}^+\mathbf{\Lambda}=\pi$, we get, using Lemma 9 and Lemma 10 on the inequality (40):

$$\begin{aligned}
\Delta_{\tilde{z}} &\triangleq \sum_{(i,j)\in\mathcal{E}}\lambda_{ij}\big(\|\tilde{z}_{ij}^+-z^*\|^2_{\mathbf{\Lambda}^+} - \|\tilde{z}-z^*\|^2_{\mathbf{\Lambda}^+}\big) &(37)\\
&= \sum_{(i,j)\in\mathcal{E}}\lambda_{ij}2\langle \tilde{z}-z^*, \tilde{z}_{ij}^+-\tilde{z}\rangle_{\mathbf{\Lambda}^+} + \|\tilde{z}_{ij}^+-\tilde{z}\|^2_{\mathbf{\Lambda}^+} &(38)\\
&= -2\tilde{\beta}\sum_{(i,j)\in\mathcal{E}}\lambda_{ij}\langle \tilde{z}-z^*, (\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}(y+z-y^*-z^*)\rangle_{\mathbf{\Lambda}^+} \\
&\quad + \sum_{(i,j)\in\mathcal{E}}\lambda_{ij}\tilde{\beta}^2\|(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}(y+z)\|^2_{\mathbf{\Lambda}^+} &(39)\\
&\leq -2\tilde{\beta}\langle \tilde{z}-z^*, \mathbf{\Lambda}^+\mathbf{\Lambda}(y+z)\rangle + \chi_2\tilde{\beta}^2\sum_{(i,j)\in\mathcal{E}}\lambda_{ij}\|(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}(y+z)\|^2 &(40)\\
&= -2\tilde{\beta}\langle \tilde{z}-z^*, \pi(y+z)\rangle + \chi_2\tilde{\beta}^2\sum_{(i,j)\in\mathcal{E}}\lambda_{ij}\|(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^\mathsf{T}(y+z)\|^2 &(41)
\end{aligned}$$

Noting that y^+, the value of y after a Poisson update, is equal to y:

$$\begin{aligned}
\Delta_z &\triangleq \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}(\|y^+ + z_{ij}^+ - y^* - z^*\|^2 - \|y + z - y^* - z^*\|^2) && (42)\\
&= 2\sum_{(i,j)\in\mathcal{E}} \lambda_{ij}\langle y + z_{ij}^+ - y - z, y + z - y^* - z^*\rangle + \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}\|y + z_{ij}^+ - y - z\|^2 && (43)\\
&= -2\sum_{(i,j)\in\mathcal{E}} \beta\lambda_{ij}\langle(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^{\mathsf{T}}(y+z), y + z - y^* - z^*\rangle \\
&\quad + \sum_{(i,j)\in\mathcal{E}} \beta^2\lambda_{ij}\|(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^{\mathsf{T}}(y+z)\|^2 && (44)\\
&= \beta(\beta - 1)\sum_{(i,j)\in\mathcal{E}} \lambda_{ij}\|(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^{\mathsf{T}}(y+z)\|^2 && (45)
\end{aligned}$$

The momentum in $\tilde{z}$ associated with $\|\tilde{z} - z^*\|^2_{\mathbf{\Lambda}^+}$ gives:

$$2\tilde{\alpha}\tilde{C}_t\langle z - \tilde{z}, \tilde{z} - z^*\rangle_{\mathbf{\Lambda}^+} \leq \tilde{\alpha}\chi_1\tilde{C}_t\|z - z^*\|^2 - \tilde{\alpha}\tilde{C}_t\|\tilde{z} - z^*\|^2_{\mathbf{\Lambda}^+} \tag{46}$$

And the one in z associated with $\|y + z - y^* - z^*\|^2$ gives:

$$2\alpha C_t\langle \tilde{z} - z, z + y - y^* - z^*\rangle \tag{47}$$

Then, assuming that $\theta\tilde{B}_t = \tilde{\beta}\tilde{C}_t = \alpha C_t$, we have:

$$\begin{aligned}
&2\alpha C_t\langle \tilde{y} - y, z + y - y^* - z^*\rangle - 2\tilde{\beta}\tilde{C}_t\langle \tilde{z} - z^*, y + z - y^* - z^*\rangle \\
&- 2\theta\tilde{B}_t\langle y + z - y^* - z^*, \tilde{y} - y^*\rangle + 2\alpha C_t\langle \tilde{z} - z, z + y - y^* - z^*\rangle && (48)\\
&= -2\alpha C_t\|y + z - y^* - z^*\|^2 && (49)
\end{aligned}$$

At this stage, we split the negative term (49) into two halves, upper-bounding one of the halves by remembering that $\frac{\nu}{L} \leq 1$ and introducing $1 \geq \tilde{\tau} > 0$:

$$\begin{aligned}
-\alpha C_t\|y + z - y^* - z^*\|^2 &\leq -\tilde{\tau}\frac{\nu}{L}\alpha C_t\|y + z - y^* - z^*\|^2 && (50)\\
&= -\tilde{\tau}\tilde{\beta}\frac{\nu}{L}\tilde{C}_t\|y + z - y^* - z^*\|^2 && (51)\\
&\leq -\tilde{\tau}\frac{1}{2}\tilde{\beta}\frac{\nu}{L}\tilde{C}_t\|z - z^*\|^2 + \tilde{\tau}\tilde{\beta}\frac{\nu}{L}\tilde{C}_t\|y - y^*\|^2 && (52)\\
&= -\tilde{\tau}\frac{1}{2}\tilde{\beta}\frac{\nu}{L}\tilde{C}_t\|z - z^*\|^2 + \tilde{\tau}\frac{\nu}{L}\frac{2\alpha\theta}{\delta}B_t\|y - y^*\|^2 && (53)
\end{aligned}$$

■

Keeping in mind that $\theta\tilde{B}_t = \tilde{\beta}\tilde{C}_t = \alpha C_t$ and $\frac{\delta}{2}\tilde{B}_t = \alpha B_t$, we put everything together. Defining $\Psi = \frac{\partial\Phi}{\partial t} + \phi_A + \phi_B + \phi_C$, we have:

$$\Psi \leq \|\nabla F(x) - \tilde{y}\|^2 \left(A_t \frac{L\gamma^2}{2} - A_t\gamma + \tilde{A}_t\tilde{\gamma}^2 + \left((\delta+\tilde{\delta})^2 - \delta\right)\tilde{B}_t \right) \quad (54)$$
$$+ \|\tilde{z} - z^*\|^2_{\mathbf{\Lambda}^+} \left(-\tilde{\alpha}\tilde{C}_t + \tilde{C}'_t\right) \quad (55)$$
$$+ \|\tilde{y} - y^*\|^2 (\tilde{B}'_t - \frac{\delta}{2}\tilde{B}_t) \quad (56)$$
$$+ \|x - x^*\|^2 (\tilde{A}_t\tilde{\eta} - \tilde{A}_t\frac{\nu\tilde{\gamma}}{2}) \quad (57)$$
$$+ \|\tilde{x} - x^*\|^2 (\tilde{A}'_t - \tilde{A}_t\tilde{\eta}) \quad (58)$$
$$+ \|\nabla F(x) - \nabla F(x^*)\|^2 (\delta\tilde{B}_t - \frac{\tilde{\gamma}}{2L}\tilde{A}_t) \quad (59)$$
$$+ \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}\|(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^{\mathsf{T}}(y+z)\|^2 \left(\chi_2\tilde{\beta}^2\tilde{C}_t + \beta(\beta-1)C_t\right) \quad (60)$$
$$+ \|z - z^*\|^2 (\chi_1\tilde{\alpha} - \tilde{\tau}\frac{1}{2}\tilde{\beta}\frac{\nu}{L})\tilde{C}_t \quad (61)$$
$$+ \|y - y^*\| (B'_t - (1 - \tilde{\tau}\frac{\nu}{L}\frac{2\theta}{\delta})\alpha B_t) \quad (62)$$
$$+ \|y + z - y^* - z^*\|^2 (C'_t - \alpha C_t) \quad (63)$$
$$+ d_F(x, x^*)(A'_t - \eta A_t) \quad (64)$$
$$+ d_F(\tilde{x}, x)(-A_t\eta + 2\tilde{\gamma}\tilde{A}_t) \quad (65)$$
$$+ d_F(\tilde{x}, x^*)(A_t\eta - 2\tilde{\gamma}\tilde{A}_t) \quad (66)$$
$$+ \langle \nabla F(x) - \tilde{y}, y^* - \tilde{y}\rangle (-2\tilde{\delta}\tilde{B}_t + \gamma A_t) \quad (67)$$
$$+ \langle \tilde{y} - y^*, \tilde{x} - x^*\rangle \left(2\tilde{\gamma}\tilde{A}_t - 2\theta\nu\tilde{B}_t\right) \quad (68)$$

Resolution **Proof** [Proof of Lemma 12] Our goal is to put to zero all of the terms appearing next to scalar products and make the factors of positive quantities (norms or divergences) less or equal to zero. Given our relations, we guess that each exponential has the same rate. Thus, with $\tau > 0$, we fix $\frac{\delta}{2} = \tilde{\eta} = \eta = \tilde{\alpha} = \tau\sqrt{\frac{\nu}{2\chi_1\chi_2 L}}$, which leads to $\tilde{\gamma} = \frac{\sqrt{2}\tau}{\sqrt{\nu L\chi_1\chi_2}}$ using Eq. (57). Also, from Eq. (66):

$$4\tilde{A}_t = \nu A_t.$$

Next, from Eq. (59) and Eq. (68), it's necessary that:

$$2L\delta = \theta\nu\,,$$

thus $\theta = 2\sqrt{2}\tau\sqrt{\frac{L}{\chi_1\chi_2\nu}}$. From Eq. (68), we get:

$$\tilde{A}_t = 2L\nu\tilde{B}_t.$$

Combining this previous equation with Eq. (67), as $4\tilde{A}_t = \nu A_t$, we have $\tilde{\delta} = 4L\gamma$. Next, Eq. (54) gives, with the equations above:

$$\begin{aligned}
A_t(\frac{L\gamma^2}{2} - \gamma) + \tilde{A}_t\tilde{\gamma}^2 + \left((\delta+\tilde{\delta})^2 - \delta\right)\tilde{B}_t &= A_t\frac{L\gamma^2}{2} - A_t\gamma + \frac{\nu}{4}\tilde{\gamma}^2 A_t + \left(\delta^2 + \tilde{\delta}^2 + \delta\right)\frac{A_t}{8L} \\
&= A_t\left(\frac{L\gamma^2}{2} - \gamma + \frac{\nu}{4}\frac{4\tau^2}{\nu L}\right) + A_t(2\tau\sqrt{\frac{\nu}{L}} + 4\tau^2\frac{\nu}{L} + 16L^2\gamma^2)\frac{1}{8L} \\
&\leq A_t(\gamma^2\frac{5}{2}L - \gamma + \frac{5}{4}\frac{\tau^2}{L} + \frac{\sqrt{2}}{8}\frac{\tau}{L})
\end{aligned}$$

We thus pick $\gamma = \frac{1}{4L}$ and $\tau = \frac{1}{8}$, so that $\tilde{\delta} = 1$. Via Eq. (62), we fix $\tilde{\tau} = \frac{1}{8} < 1$. With Eq. (61), we then get:

$$\tilde{\beta} = \sqrt{2\frac{\chi_1 L}{\chi_2 \nu}}$$

We also put $\alpha = \tau\sqrt{\frac{\nu}{2L\chi_1\chi_2}}$ and only one last equation, Eq. (60), needs to be satisfied, for which we pick $\beta = \frac{1}{2}$:

$$\chi_2\tilde{\beta}^2\tilde{C}_t + \beta(\beta-1)C_t = (\chi_2\tilde{\beta}\alpha - \frac{1}{4})C_t = (2\tau - \frac{1}{4})C_t = 0$$

In summary, we set:

$$\begin{array}{llll}
\eta = \frac{1}{8\sqrt{2}}\sqrt{\frac{\nu}{\chi_1\chi_2 L}} & \gamma = \frac{1}{4L} & \delta = \frac{1}{4\sqrt{2}}\sqrt{\frac{\nu}{\chi_1\chi_2 L}} & \alpha = \frac{1}{8\sqrt{2}}\sqrt{\frac{\nu}{\chi_1\chi_2 L}} \\
\beta = \frac{1}{2} & \theta = \frac{1}{2\sqrt{2}}\sqrt{\frac{L}{\chi_1\chi_2\nu}} & \tilde{\tau} = \frac{1}{8} & \tilde{\eta} = \frac{1}{8}\sqrt{\frac{\nu}{L}} \\
\tilde{\gamma} = \frac{1}{4\sqrt{2}\sqrt{\nu L\chi_1\chi_2}} & \tilde{\delta} = 1 & \tilde{\alpha} = \frac{1}{8\sqrt{2}}\sqrt{\frac{\nu}{\chi_1\chi_2 L}} & \tilde{\beta} = \sqrt{2\frac{\chi_1 L}{\chi_2\nu}} \\
\nu = \frac{\mu}{2} & \tau = \frac{1}{8} & &
\end{array}$$

Now, let's pick: $A_t = e^{t\tau\sqrt{\frac{\nu}{2L\chi_1\chi_2}}} = e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}}$ so that:

$$\begin{array}{lll}
A_t = e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} & \tilde{A}_t = \frac{\mu}{8}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} & \tilde{B}_t = \frac{1}{8L}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} \\
B_t = \frac{1}{16L}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} & C_t = \frac{1}{\mu}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}} & \tilde{C}_t = \frac{1}{32\chi_1 L}e^{\frac{t}{8\sqrt{2}}\sqrt{\frac{\mu}{L\chi_1\chi_2}}}.
\end{array}$$

This implies that $\Psi \leq 0$.

■

Appendix D. Computation of graph relevant quantities for standard graphs

All these graphs have permutation symmetries, and we assume them to be undirected. Consequently, the graph resistance χ_2 is constant, and the graph weights λ are also constant (normalization choices should not affect computations, so we will pick $\lambda = 1$). Therefore, Eq. 10 becomes an equality:

$$n - 1 = 2\lambda|\bar{\mathcal{E}}|\chi_2 \quad \text{and} \quad \operatorname{Tr}\Lambda = 2\lambda|\bar{\mathcal{E}}|.$$

Line graph. Recall that the Laplacian Λ_{line} of a line graph has a Toeplitz structure and is diagonalizable by the Discrete Fourier Transform. The eigenvalues of the Laplacian are given by $\{4\sin^2\left(\frac{\pi k}{2(n+1)}\right)\}_{k\leq n}$, and hence, $\chi_1 \sim \frac{1}{\pi}n^2$. Here, $|\bar{\mathcal{E}}| = 2n$, so $\chi_2 = \frac{n-1}{2n}$, $\|\Lambda\| = 4$, and $\operatorname{Tr}\Lambda = 2n$.

***d*-Grid graph.** Let $n = m^d$ for some $m \in \mathbb{N}$. A d-dimensional grid is the tensor sum of d line graphs, yielding a Laplacian of the form

$$\Lambda = \sum_{i=1}^{d} I^{\otimes(i-1)} \otimes \Lambda_{\text{line}} \otimes I^{\otimes(d-i)}.$$

The spectrum is thus given by:

$$\left\{4\sum_{i=1}^{d}\sin^2\left(\frac{\pi k_i}{2(m+1)}\right) \,:\, 1 \leq k_i \leq m\right\}.$$

Thus, $\chi_1 \sim \frac{m^2}{\pi} \sim \frac{n^{\frac{2}{d}}}{\pi}$, and $\|\Lambda\| = 4d$. The total number of edges is $|\bar{\mathcal{E}}| = dn$, giving $\operatorname{Tr}\Lambda = 2dn$ and $\chi_2 = \frac{(n-1)}{2dn}$.

Complete graph. The spectrum consists of a zero eigenvalue and a repeated eigenvalue $\frac{1}{\chi_1}$ with multiplicity $n-1$. Thus: $\chi_2 = \frac{n-1}{2n^2}$, $\quad \operatorname{Tr}\Lambda = 2n^2$, $\quad \chi_1 = \frac{1}{\|\Lambda\|} = \frac{n-1}{2n^2}$.

Star graph. Here, the Laplacian is:

$$\Lambda = \sum_{i=2}^{n}(e_i - e_1)(e_i - e_1)^{\mathsf{T}}.$$

Assuming all nodes are connected to a central node, it's easy to see:

$$\Lambda\left(e_1 - \frac{1}{n}\sum_{i=1}^{n} e_i\right) = n\left(e_1 - \frac{1}{n}\sum_{i=1}^{n} e_i\right),$$

and for all $n-1 \geq i \geq 2, \Lambda(e_{i+1} - e_i) = e_{i+1} - e_i$ Thus, we recover the full spectrum, giving since $|\bar{\mathcal{E}}| = n-1$:

$$\chi_2 = \frac{1}{2}, \quad \operatorname{Tr}\Lambda = 2(n-1), \quad \chi_1 = 1, \quad \|\Lambda\| = n.$$

Appendix E. Comparison of complexities with related work

Proof [Proof of Prop. 5] We consider the settings of concurrent works and, given any gossip matrix admissible for them, we show that DADAO has better rates.

Comparison with ADOM+ (Kovalev et al., 2021a). The ADOM+ setting is:

- *Gossip matrices*: Laplacians $\mathcal{L}$ with $\|\mathcal{L}x - x\|^2 \leq (1 - \frac{1}{\chi})\|x\|^2$ for $\chi \geq 1$ and $x \in \mathbf{1}^\perp$.
- *Total number of gradients to reach ϵ precision*: $\mathcal{O}(n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$.
- *Total number of edges activated to reach ϵ precision*: $\mathcal{O}(|\mathcal{E}|\chi\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$.

If we take an eigenvector of $\mathcal{L}$ for the eigenvalue $\frac{1}{\chi_1[\mathcal{L}]}$, then the Laplacian inequality directly leads to $\left(1 - \frac{1}{\chi_1[\mathcal{L}]}\right)^2 \leq 1 - \frac{1}{\chi}$ and we have:

$$\frac{1}{\chi_1[\mathcal{L}]}\left(\frac{1}{\chi_1[\mathcal{L}]} - 2\right) \leq -\frac{1}{\chi},$$

leading to:

$$\frac{2}{\chi_1[\mathcal{L}]} \geq \frac{1}{\chi_1[\mathcal{L}]}\left(2 - \frac{1}{\chi_1[\mathcal{L}]}\right) \geq \frac{1}{\chi}.$$

Thus, $\chi_1[\mathcal{L}] \leq 2\chi$. Remind that, by definition, $\chi_2 \leq \chi_1$. Note that in ADOM+, we also have $\|\mathcal{L}\| \leq 2$, so that $\mathrm{Tr}(\mathcal{L}) \leq 2n$. Then, for any Laplacian matrix $\mathcal{L}$ valid for ADOM+, we consider for DADAO a Λ defined as followed:

$$\Lambda = \sqrt{2\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]}\mathcal{L}$$

Then, DADAO has the same gradient complexity as ADOM+, but the number of communications of DADAO is:

$$\frac{T}{2}\mathrm{Tr}\,\Lambda \leq \frac{1}{2}\sqrt{2\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]}\sqrt{\frac{L}{\mu}}\log\left(\frac{1}{\epsilon}\right)2n = \mathcal{O}(|\mathcal{E}|\chi\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$$

Consequently, DADAO is better than ADOM+ for all valid configurations of ADOM+ (in the fixed graph topology setting) and DADAO. We note that for the complete graph, $\chi = \mathcal{O}(1)$ and $|\mathcal{E}| = \mathcal{O}(n^2)$, whereas $n\sqrt{2\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]} = \mathcal{O}(n)$ and DADAO has strictly better communication rates than ADOM+ on this graph.

Comparison with Gradient Tracking methods AGT, OGT (Li and Lin, 2021; Song et al., 2021). The setting is described by:

- *Gossip matrices*: Any matrix $\mathcal{L} = \mathbf{I} - W$ with W symmetric doubly-stochastic, *i.e.* such that $\sum_i W_{ij} = 1$ and $\sum_j W_{ij} = 1$.
- *Total number of gradients to reach ϵ precision*: $\mathcal{O}(n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$.
- *Total number of edges activated to reach ϵ precision*: $\mathcal{O}(|\mathcal{E}|\frac{1}{\sqrt{\theta}}\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$, where $\theta = 1 - \|W - \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}\|$.

If $\kappa = \mathcal{O}(\frac{|\mathcal{E}|}{n})$, using Lemma 2, we have:

$$\frac{1}{2}\sqrt{2\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]}\mathrm{Tr}\,\mathcal{L} = \mathcal{O}(|\mathcal{E}|\sqrt{\rho[\mathcal{L}]})$$

Furthermore, as W is stochastic, $\|\mathcal{L}\| \leq 2$ and $\theta \leq \frac{1}{\chi_1[\mathcal{L}]}$, leading to $\rho \leq \frac{2}{\theta}$. Consequently, the communication complexity of DADAO run for $\mathcal{O}(\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$ iterations recovers the rate of GT. Furthermore, for the complete graph, for any Laplacian $\mathcal{L}$ with $\rho[\mathcal{L}] = \mathcal{O}(1)$ (remind that, by definition $\rho \geq 1$), we have $\sqrt{2\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]}\mathrm{Tr}\,\mathcal{L} = \mathcal{O}(n)$ whereas $|\mathcal{E}| = \mathcal{O}(n^2)$, thus DADAO uses an order of magnitude less communications than GT need to.

Note that for the Star-graph, there is no admissible Laplacian in the framework of Song et al. (2021).

Comparison with MSDA (Scaman et al., 2017). The setting is described by:

- *Gossip matrices*: any Laplacians admissible for DADAO.
- *Total number of gradients to reach ϵ precision*: $\mathcal{O}(n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$.
- *Total number of edges activated to reach ϵ precision*: $\mathcal{O}(|\mathcal{E}|\sqrt{\rho}\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$.

If $\kappa = \mathcal{O}(\frac{|\mathcal{E}|}{n})$, using Lemma 2, we see that, for any Laplacian $\mathcal{L}$:

$$\frac{1}{2}\sqrt{2\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]}\mathrm{Tr}\,\mathcal{L} = \mathcal{O}(|\mathcal{E}|\sqrt{\rho})$$

Consequently, the communication complexity of DADAO run for $\mathcal{O}(\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$ iterations is better than MSDA. Furthermore, for the complete graph, for any Laplacian $\mathcal{L}$ with $\rho[\mathcal{L}] = \mathcal{O}(1)$ (remind that, by definition $\rho \geq 1$), we have $\sqrt{2\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]}\mathrm{Tr}\,\mathcal{L} = \mathcal{O}(n)$ whereas $|\mathcal{E}| = \mathcal{O}(n^2)$, thus DADAO uses an order of magnitude less communications than MSDA need to.

Comparison with the Continuized framework (Even et al., 2021a). In this framework and using their notations, we have:

- *Gossip matrices*: all Laplacian matrices $\mathcal{L}$ verifying $\mathrm{Tr}\mathcal{L} = 2$.
- *Total number of gradients to reach ϵ precision*: $\mathcal{O}(\frac{1}{\theta'_{\mathrm{ARG}}}\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$.
- *Total number of edges activated to reach ϵ precision*: $\mathcal{O}(\frac{1}{\theta'_{\mathrm{ARG}}}\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$.

First, we slightly rephrase one of the proposition of Even et al. (2021a) to match our notations:

Lemma 16 *For a Laplacian $\mathcal{L}$ with $\mathrm{Tr}\,\mathcal{L} = 2$, the communication and gradient complexity of the continuized framework is given by:*

$$\mathcal{O}(\sqrt{\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]}\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}).$$

Proof Under the notation of Even et al. (2021a), $\theta'_{\text{ARG}} = \sqrt{\mu_{\text{gossip}} / \max_{\{v,w\}} \frac{R_{vw}}{P_{vw}}}$, with $\mu_{\text{gossip}} = \frac{1}{\chi_1[\mathcal{L}]}$ in our setting. Moreover, we remind that in Even et al. (2021a), $\mathcal{L} = AA^{\mathsf{T}}$ and that $Ae_{vw} = \sqrt{P_{vw}}(e_v - e_w)$. Thus, by definition:

$$\frac{R_{vw}}{P_{vw}} \triangleq \frac{e_{vw}^{\mathsf{T}} A^+ A e_{vw}}{P_{vw}} \tag{69}$$
$$= \frac{e_{vw}^{\mathsf{T}} A^+ (e_v - e_w)}{\sqrt{P_{vw}}} \tag{70}$$
$$= \frac{(A^{+\mathsf{T}} e_{vw})^{\mathsf{T}} (e_v - e_w)}{\sqrt{P_{vw}}} \tag{71}$$
$$= \frac{((AA^{\mathsf{T}})^{+\mathsf{T}} A e_{vw})^{\mathsf{T}} (e_v - e_w)}{\sqrt{P_{vw}}} \tag{72}$$
$$= (e_v - e_w)^{\mathsf{T}} \mathcal{L}^+ (e_v - e_w)\,, \tag{73}$$

and we have $\theta'_{\text{ARG}} = \frac{1}{\sqrt{2\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]}}$. ■

Next, if $\operatorname{Tr}\mathcal{L} = 2$, we proved in Lemma 2that $2\sqrt{\chi_1[\mathcal{L}]\chi_2[\mathcal{L}]} \geq (n-1)$. Thus, we see that the gradient complexity of DADAO in $\mathcal{O}(n\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon})$ is always better than the one of the continuized framework. If we write $f(n) = \Omega(g(n))$ the fact that there is a constant $C > 0$ and $n_0 \in \mathbb{N}$ such that $\forall n \geq n_0,\ Cg(n) \leq f(n)$, then, in the cycle graph, for any $\mathcal{L}$ with $\operatorname{Tr}\mathcal{L} = 2$, $\chi_1[\mathcal{L}] = \Omega(n^3)$ and $\chi_2[\mathcal{L}] = \Omega(n)$. Thus DADAO uses an order of magnitude less gradients than the continuized framework for this graph.

■

Appendix F. Optimizing the communication rates with a SDP

We remind the following useful Lemma:

Lemma 17 (Generalized Schur complement) *Let M be a symmetric matrix defined as:*

$$M = \begin{pmatrix} A & B \\ B^{\mathsf{T}} & C \end{pmatrix}.$$

Then:

- $M \succcurlyeq 0 \Leftrightarrow A \succ 0 \quad , \quad C - B^{\mathsf{T}} A^{-1} B \succcurlyeq 0.$
- $M \succcurlyeq 0 \Leftrightarrow A \succcurlyeq 0 \quad , \quad (\mathbf{I} - AA^+)B = 0 \quad , \quad C - B^{\mathsf{T}} A^+ B \succcurlyeq 0.$

F.1 Convexity lemmas

Next, we prove our convexity bounds:

Proof [Proof of Lemma 6] We want to show that $\lambda \to \psi(\lambda, u)$ is log-convex on $\mathbb{R}_+^{n^2}$ and strictly convex on the convex set $\{\lambda, \chi_1(\Lambda(\lambda)) < \infty\}$ of the connected graphs. Having in mind that:

$$\frac{d}{dt}X^+(t) = -X^+(t)\frac{d}{dt}X(t)X^+(t)\,,$$

we note that:

$$\frac{\partial}{\partial \lambda_{ij}} \log \psi(\Lambda) = -\frac{u^\mathsf{T} \Lambda^+ (e_i - e_j)(e_i - e_j)^\mathsf{T} \Lambda^+ u}{u^\mathsf{T} \Lambda^+ u} \tag{74}$$

$$= -\frac{(u^\mathsf{T} \Lambda^+ (e_i - e_j))^2}{u^\mathsf{T} \Lambda^+ u} \tag{75}$$

and next:

$$\frac{\partial^2}{\partial \lambda_{ij} \partial \lambda_{i'j'}} \log \psi(\Lambda) = \frac{2u^\mathsf{T} \Lambda^+ (e_i - e_j) u^\mathsf{T} \Lambda^+ (e_{i'} - e_{j'})(e_{i'} - e_{j'})^\mathsf{T} \Lambda^+ (e_i - e_j) u^\mathsf{T} \Lambda^+ u}{(u^\mathsf{T} \Lambda^+ u)^2} - \frac{(u^\mathsf{T} \Lambda^+ (e_i - e_j))^2 (u^\mathsf{T} \Lambda^+ (e_{i'} - e_{j'}))^2}{(u^\mathsf{T} \Lambda^+ u)^2} \tag{76}$$

Now, let the $(\delta_{ij})_{1 \leq i,j \leq n}$ be positive and introduce $\Delta = \sum_{ij} \delta_{ij} (e_i - e_j)(e_i - e_j)^\mathsf{T}$. Then, with $\mathcal{H}(\psi)$ the hessian of $\log \psi$, we observe that:

$$\Delta^\mathsf{T} \mathcal{H}(\psi) \Delta = \sum_{ij, i'j'} \delta_{ij} \delta_{i'j'} \frac{\partial^2}{\partial \lambda_{ij} \partial \lambda_{i'j'}} \psi(\Lambda) \tag{77}$$

$$= \frac{2u^\mathsf{T} \Lambda^+ \Delta \Lambda^+ \Delta \Lambda^+ u (u^\mathsf{T} \Lambda^+ u) - (u^\mathsf{T} \Lambda^+ \Delta \Lambda^+ u)^2}{(u^\mathsf{T} \Lambda^+ u)^2} \tag{78}$$

As $u^\mathsf{T} \Lambda^+ \Delta \Lambda^+ \Delta \Lambda^+ u (u^\mathsf{T} \Lambda^+ u) \geq 0$, $\Delta^\mathsf{T} \mathcal{H}(\psi) \Delta$ would be positive if we show

$$u^\mathsf{T} \Lambda^+ \Delta \Lambda^+ \Delta \Lambda^+ u (u^\mathsf{T} \Lambda^+ u) - (u^\mathsf{T} \Lambda^+ \Delta \Lambda^+ u)^2 \geq 0\,. \tag{79}$$

As $u \perp \mathbf{1}$, by Schur complement, this would be the case if:

$$\begin{pmatrix} u^\mathsf{T} \Lambda^+ \Delta \Lambda^+ \Delta \Lambda^+ u & u^\mathsf{T} \Lambda^+ \Delta \Lambda^+ u \\ u^\mathsf{T} \Lambda^+ \Delta \Lambda^+ u & u^\mathsf{T} \Lambda^+ u \end{pmatrix} \succcurlyeq 0\,, \tag{80}$$

but this also writes:

$$\begin{pmatrix} u \\ u \end{pmatrix}^\mathsf{T} \begin{pmatrix} \Lambda^+ \Delta \Lambda^+ \Delta \Lambda^+ & \Lambda^+ \Delta \Lambda^+ \\ \Lambda^+ \Delta \Lambda^+ & \Lambda^+ \end{pmatrix} \begin{pmatrix} u \\ u \end{pmatrix} \succcurlyeq 0\,. \tag{81}$$

For this to be true for any u, by Schur complement, this would be implied by:

$$\Lambda^+ \succcurlyeq 0 \quad , \quad (\mathbf{I} - \Lambda^+ \Lambda) \Lambda^+ \Delta \Lambda^+ = 0$$

and

$$\Lambda^+ \Delta \Lambda^+ \Delta \Lambda^+ - \Lambda^+ \Delta \Lambda^+ \Lambda \Lambda^+ \Delta \Lambda^+ \succcurlyeq 0\,.$$

As $\Lambda^+ \Lambda = \pi$, all conditions are met. In particular, (79) and having:

$$\Delta^\mathsf{T} \mathcal{H}(\psi) \Delta = 0$$

would imply that:

$$u^\mathsf{T} \Lambda^+ \Delta \Lambda^+ \Delta \Lambda^+ u u^\mathsf{T} \Lambda^+ u = 0$$

As $u \neq 0$, if Δ represents a connected graph, this can never be equal to 0. ■

Corollary 18 (Strict convexity of the relaxed communication rate) *Fix a directed graph $\mathcal{E}$ so that $\bar{\mathcal{E}}$ is connected. Then, $\lambda \to \sqrt{\chi_1[\Lambda(\lambda)]\chi_2^{\mathcal{E}}(\lambda)}$ is strictly convex on $\mathbb{R}_+^{\mathcal{E}}$.*

Proof [Proof of Corollary 18] We note that the supremum of logarithmic convex functions is logarithmically convex, thus $\lambda \to \chi_1[\Lambda(\lambda)], \lambda \to \chi_2[\Lambda(\lambda)]$ are logarithmically convex and so is their square root. Furthermore, $\lambda \to \chi_2[\Lambda(\lambda)]$ is a finite supremum (ie, a maximum) of strictly convex function, thus it is strictly convex and so is $\lambda \to \sqrt{\chi_1[\Lambda(\lambda)]\chi_2[\Lambda(\lambda)]}$. ■

F.2 Problem relaxation

We want to consider the following:

$$\begin{aligned} \text{minimize} \quad & \operatorname{Tr}\Lambda(\lambda)\sqrt{2\chi_1[\Lambda(\lambda)]\chi_2[\Lambda(\lambda)]} \\ \text{subject to} \quad & \lambda \in \mathcal{C} \cap \mathbb{R}_+^{\mathcal{E}}\,. \end{aligned} \tag{82}$$

As shown in the two-dimensional case, since the function $(x, y) \mapsto (x + y)\sqrt{\min\left(\frac{1}{x}, \frac{1}{y}\right)}$ is not convex, optimizing the preceding expression requires additional assumptions. For $\mathcal{C}_1$, by homogeneity (the quantity of interest is invariant by rescaling), the optimal solution can be picked as equality in the inequality constraints. For $\mathcal{C}_2, \mathcal{C}_3$, we will simplify Eq. 82 using the fact that:

$$\operatorname{Tr}\Lambda(\lambda) = 2\sum_{i=1}^{n}\sum_{j,(ij)\in\mathcal{E}} \lambda_{ij} \leq 2\sum_{i=1}^{n} \lambda_i\,. \tag{83}$$

This gives an upper bound on the trace, which allows us to consider a relaxed formulation by dropping the trace in Eq. (82). However, this comes at a cost: the relaxed objective no longer accounts for the trade-offs between saturating a few links versus distributing bandwidth more evenly across the graph. This can lead to solutions that over-emphasize certain edges, especially in graphs with inherent bottlenecks (e.g., the Barbell graph), where the optimal allocation under local constraints would focus bandwidth on the connecting edge alone. Our problem thus becomes:

$$\begin{aligned} \text{minimize} \quad & \sqrt{2\chi_1[\Lambda(\lambda)]\chi_2[\Lambda(\lambda)]} \\ \text{subject to} \quad & \lambda \in \mathcal{C} \cap \mathbb{R}_+^{\mathcal{E}}\,. \end{aligned} \tag{84}$$

However, $\lambda \to \chi_2[\Lambda(\lambda)]$ is not convex as explained above. Hence, we relax the problem by considering:

$$\begin{aligned} \text{minimize} \quad & \sqrt{2\chi_1[\Lambda(\lambda)]\chi_2^{\mathcal{E}}(\lambda)} \\ \text{subject to} \quad & \lambda \in \mathcal{C} \cap \mathbb{R}_+^{\mathcal{E}}\,. \end{aligned} \tag{85}$$

Remark 19 *We note that we have here a Braess paradox (Frank, 1981): minimizing Eq.* (84) *compared to minimizing Eq.* (85) *can lead to a substantially better communication rate.*

However, finding a solution to problem (84) *requires in practice knowing the support of the optimal graph in advance. If some edges are dropped during the optimization procedure (* i.e., *allocated with a 0 weight), minimizing Eq.* (85) *with the optimal support would lead to the optimal communication rate.*

F.3 SDP formulation

Proof [Proof of Prop. 7] First, note that $\varphi(u) \triangleq ut_1 + \frac{t_2}{u}$ satisfies $\varphi'(u) = t_1 - \frac{t_2}{u^2}$ so that the minimum is reached in $u^* = \sqrt{\frac{t_2}{t_1}}$, with $\varphi(u^*) = 2\sqrt{t_1 t_2}$. Now, if $t_2 \geq t_1$, $u^* \geq 1$ and we note that $\varphi(u^*) = \inf_{u \geq 1} \varphi(u)$.

Our initial problem writes:

$$\begin{array}{rl} \text{minimize} & \sqrt{2\|\Lambda^+\| \frac{1}{2} \sup_{(ij)\in\mathcal{E}} (e_i - e_j)^\mathsf{T} \Lambda^+ (e_i - e_j)} \\ \text{subject to} & \lambda \in \mathcal{C}, \lambda_{ij} \geq 0, \forall (i,j) \in \mathcal{E} \\ & \Lambda \ = \ \sum_{ij} \lambda_{ij} (e_i - e_j)(e_i - e_j)^\mathsf{T} \end{array}$$

Thus, as $\chi_2 \leq \chi_1$, it is equivalent to solving:

$$\begin{array}{rl} \text{minimize} & u \sup_{(ij)\in\mathcal{E}} (e_i - e_j)^\mathsf{T} \Lambda^+ (e_i - e_j) + \frac{\|\Lambda^+\|}{u} \\ \text{subject to} & \lambda \in \mathcal{C}, u \geq 1, \lambda_{ij} \geq 0, \forall (i,j) \in \mathcal{E} \\ & \Lambda \ = \ \sum_{ij} \lambda_{ij} (e_i - e_j)(e_i - e_j)^\mathsf{T} \end{array}$$

which also writes:

$$\begin{array}{rl} \text{minimize} & ut_1 + \frac{t_2}{u} \\ \text{subject to} & \lambda \in \mathcal{C}, u \geq 1, \lambda_{ij} \geq 0, \forall (i,j) \in \mathcal{E} \\ & \Lambda \ = \ \sum_{ij} \lambda_{ij} (e_i - e_j)(e_i - e_j)^\mathsf{T} \\ & t_1 \ \geq \ (e_i - e_j)^\mathsf{T} \Lambda^+ (e_i - e_j) \quad \forall (i,j) \in \mathcal{E} \\ & t_2 \ \geq \ \|\Lambda^+\| \end{array}$$

equivalent to (with $\tilde{\lambda}_{ij} = u\lambda$ so that $\tilde{\Lambda} = u\Lambda$, $\tilde{\Lambda}^+ = \frac{1}{u}\Lambda^+$ and $\tilde{\lambda} \in u\mathcal{C}$):

$$\begin{array}{rl} \text{minimize} & ut_1 + \frac{t_2}{u} \\ \text{subject to} & \tilde{\lambda} \in u\mathcal{C}, u \geq 1, \tilde{\lambda}_{ij} \geq 0, \forall (i,j) \in \mathcal{E} \\ & \tilde{\Lambda} \ = \ \sum_{ij} \tilde{\lambda}_{ij} (e_i - e_j)(e_i - e_j)^\mathsf{T} \\ & t_1 \ \geq \ u(e_i - e_j)^\mathsf{T} \tilde{\Lambda}^+ (e_i - e_j) \quad \forall (i,j) \in \mathcal{E} \\ & t_2 \ \geq \ u\|\tilde{\Lambda}^+\| \end{array}$$

equivalent to (with $t_1' = ut_1, t_2' = \frac{t_2}{u}$):

$$\begin{array}{rl}\text{minimize} & t_1' + t_2' \\ \text{subject to} & \lambda \in u\mathcal{C}, u \geq 1, \lambda_{ij} \geq 0, \forall (i,j) \in \mathcal{E} \\ & \Lambda \;=\; \sum_{ij} \lambda_{ij}(e_i - e_j)(e_i - e_j)^\intercal \\ & t_1' \;\geq\; u^2 (e_i - e_j)^\intercal \Lambda^+ (e_i - e_j) \quad \forall (i,j) \in \mathcal{E} \\ & t_2' \;\geq\; \|\Lambda^+\| \end{array}$$

equivalent to, via Schur Complement:

$$\begin{array}{rl}\text{minimize} & t_1 + t_2 \\ \text{subject to} & \lambda \in u\mathcal{C}, u \geq 1, \lambda_{ij} \geq 0, \forall (i,j) \in \mathcal{E} \\ & \Lambda = \sum_{ij} \lambda_{ij}(e_i - e_j)(e_i - e_j)^\intercal \\ & \begin{pmatrix} \Lambda & u(e_i - e_j) \\ u(e_i - e_j)^\intercal & t_1 \end{pmatrix} \succcurlyeq 0 \quad \forall (i,j) \in \mathcal{E} \\ & \begin{pmatrix} \Lambda & \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\intercal \\ \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\intercal & t_2 \mathbf{I} \end{pmatrix} \succcurlyeq 0 \end{array}$$

■

F.4 Optimal weights for the Barbell Graph

Proof [Proof of Lemma 8] Via a symmetry argument similar to Ghosh et al. (2008), we know that the problem actually depends on three variables a, b, c.

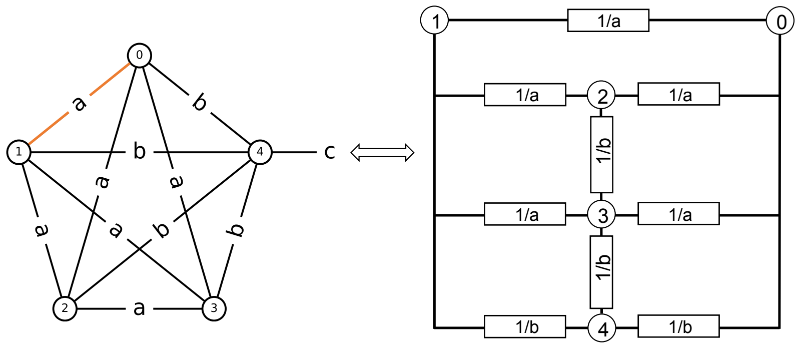


Figure 6: Equivalent resistance network if some current passes between node 0 and node 1 of the Barbell graph $K_5 - K_5$. By symmetry of the resistance network, no current passes through the edges $2-3$ and $3-4$ and we have $\frac{1}{2R_a} = a + \frac{2}{\frac{1}{a} + \frac{1}{a}} + \frac{1}{\frac{1}{b} + \frac{1}{b}} = 2a + \frac{b}{2}$.

By symmetry and using the Ohm law, we know that the resistance of the corresponding edges are given by:

$$2R_c = \frac{1}{c}, \frac{1}{2R_a} = a + \frac{a}{2}(n-3) + \frac{b}{2} = \frac{1}{2}(a(n-1)+b), \frac{1}{2R_b} = b + \frac{1}{\frac{1}{a}+\frac{1}{b}}(n-2)\,,$$

see Fig. 6 for an example of such computation on the Barbell graph $K_5 - K_5$. We can simplify the expressions as follows:

$$R_a = \frac{1}{a(n-1)+b} \quad , \quad R_b = \frac{a+b}{2b}R_a \quad , \quad R_c = \frac{1}{2c}\,.$$

Directly using the results of Ghosh et al. (2008), the eigenvalues of the graph's Laplacian are:

$$a(n-1)+b, nb, \frac{1}{2}(nb + 2c \pm \sqrt{(2c+nb)^2 - 8bc})$$

and the global bandwidth constraint $\mathcal{C}_1$ translates into:

$$1 = c + 2(n-1)b + (n-1)(n-2)a\,.$$

From the drawing, it is natural that $a^* \leq b^* \leq c^*$.

Finding the effective resistance of the graph:
We want to find $\chi_2^* = \max(R_{a^*}, R_{b^*}, R_{c^*})$. We can directly eliminate R_{b^*} from the list as $b^* \geq a^*$, leading to $R_{b^*} \leq R_{a^*}$. From the constraints, we have:

$$\frac{1}{n-1} = \frac{c^*}{n-1} + 2b^* + (n-2)a^* \geq b^* + (n-1)a^*\,,$$

meaning $R_{a^*} \geq n-1$. Thus, if we assume from the drawing $c^* = \mathcal{O}(1)$, then $R_{c^*} = \mathcal{O}(1)$ and $\chi_2^* = R_{a^*}$. Let us make this assumption for now.

Finding the inverse of the smallest eigenvalue of the Laplacian:
We want to find $\frac{1}{\chi_1^*} = \min\left(a^*(n-1)+b^*, nb^*, \frac{1}{2}(nb^* + 2c^* \pm \sqrt{(2c^*+nb^*)^2 - 8b^*c^*})\right)$. As $b^* \geq a^*$ and for obvious reasons, we directly have:

$$\frac{1}{\chi_1^*} = \min\left(a^*(n-1)+b^*, \frac{1}{2}(nb^* + 2c^* - \sqrt{(2c^*+nb^*)^2 - 8b^*c^*})\right)\,.$$

For now, we will also assume that $\chi_2^* < \chi_1^*$, and as $a^*(n-1)+b^* = \frac{1}{\chi_2^*}$, this means $\frac{1}{\chi_1^*} = \frac{1}{2}(nb^* + 2c^* - \sqrt{(2c^*+nb^*)^2 - 8b^*c^*})$.

Solving the system:
Now, we define $f(a,b,c) = \frac{1}{2\chi_1(a,b,c)\chi_2(a,b,c)}$. We want to find a^*, b^*, c^* solving the following:

$$\begin{aligned} \text{maximize} \quad & f(a,b,c) = (a(n-1)+b)\left(nb + 2c - \sqrt{(2c+nb)^2 - 8bc}\right) \\ \text{subject to} \quad & 1 = c + 2(n-1)b + (n-1)(n-2)a \end{aligned}$$

First, from the constraints, we have $a = \frac{2c}{(n-1)(n-2)} - \frac{2b}{n-2}$. Substituting a in $f(a,b,c)$ leads to:

$$f(a,b,c) = \left(\frac{1-c-bn}{n-2}\right)\left(nb + 2c - \sqrt{(2c+nb)^2 - 8bc}\right).$$

Differentiating with respect to b and c leads to:

$$\frac{\partial f}{\partial b} = 0 \Leftrightarrow n\left(nb + 2c - \sqrt{(2c+nb)^2 - 8bc}\right) = (1-c-bn)\left(n - \frac{n(2c+nb) - 4c}{\sqrt{(2c+nb)^2 - 8bc}}\right)$$
$$\frac{\partial f}{\partial c} = 0 \Leftrightarrow \left(nb + 2c - \sqrt{(2c+nb)^2 - 8bc}\right) = (1-c-bn)\left(2 - \frac{2(2c+nb) - 4b}{\sqrt{(2c+nb)^2 - 8bc}}\right)$$

Subtracting n times the second line to the first leads to the following system:

$$\begin{cases} \left(nb + 2c - \sqrt{(2c+nb)^2 - 8bc}\right) & = (1-c-bn)\left(n - \frac{n(2c+nb)-4c}{\sqrt{(2c+nb)^2-8bc}}\right) \\ \sqrt{(2c+nb)^2 - 8bc} & = 2c + nb + 4\left(\frac{c}{n} - b\right) \end{cases}$$

Assuming $n > 3$, solving the system leads to:

$$c^* = \frac{4 - n + \sqrt{2n(n-1)}}{2(8+n)} \underset{n\to\infty}{\sim} \frac{\sqrt{2}-1}{2}$$
$$b^* = \frac{c^*}{n-2}\left(\sqrt{\frac{2(n-1)}{n}} - \frac{2}{n}\right) \underset{n\to\infty}{\sim} \frac{\sqrt{2}-1}{n\sqrt{2}}$$
$$a^* = \frac{1 - c^* - 2b^*(n-1)}{(n-1)(n-2)} \underset{n\to\infty}{\sim} \frac{\sqrt{2}-1}{2n^2}.$$

We verify that indeed $c^* = \mathcal{O}(1)$ and that $\chi_2^* < \chi_1^*$, validating our assumptions. Finally, we compute:

$$\sqrt{2\chi_1^*\chi_2^*} = \frac{1}{\sqrt{f(a^*,b^*,c^*)}} = 2(\sqrt{2}-1)\sqrt{\frac{2(2+\sqrt{2})}{80 - 56\sqrt{2}}n} + \mathcal{O}(1).$$

The case for uniform weights:
Now, we assume $a = b = c = \frac{1}{|\mathcal{E}|}$. Then, we necessarily have $\chi_2 = \frac{1}{2c} = \frac{|\mathcal{E}|}{2} = \frac{n(n-1)+1}{2}$. We also evidently have $\frac{1}{\chi_1} = \frac{1}{2}(nb + 2c - \sqrt{(2c+nb)^2 - 8bc}) = \frac{c}{2}\left(n + 2 - \sqrt{(n+2)^2 - 8}\right)$. Thus, in the end:

$$\sqrt{2\chi_1\chi_2} = (n(n-1)+1)\sqrt{\frac{2}{n+2-\sqrt{(n+2)^2-8}}} = \frac{n^{5/2}}{\sqrt{2}} + \mathcal{O}(\sqrt{n}).$$

■

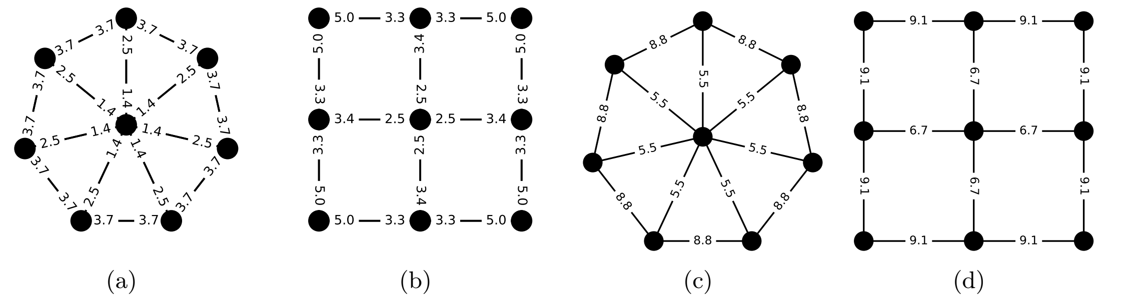


Figure 7: Optimal communication rates for the problem 85 obtained by solving the SDP from Prop. 7, under constraints $\forall i$, $\sum_{j,(i,j)\in\mathcal{E}} \lambda_{ij} \leq 1$ for (a,b) and $\sum_{(i,j)\in\mathcal{E}} \lambda_{ij} \leq 1$ for (c,d). For the sake of readability, we multiplied the values by 10 in (a,b), and by 100 in (c,d).

F.5 Results from our SDP solver

Efficient SDP solvers exist (Diamond and Boyd, 2016), and we display on Fig. 7 several results obtained thanks to the formulation of Prop. 7. Note that the solutions in Fig. 7(a,b) differ from what the simple heuristic of weighting each directed edge (i,j) with its degree $\frac{1}{d_i}$ would give or what would be prescribed with Metropolis weights, *i.e.* using $\frac{1}{1+\max\{d_i,d_j\}}$ (Xiao et al., 2007; Shi et al., 2015; Nedic et al., 2017; Gorbunov et al., 2022). Also, solutions in Fig. 7(c,d) do not follow the naive approach of putting uniform weights $\frac{1}{|\mathcal{E}|}$ on every edge.

Appendix G. Physical interpretation of the sufficient condition

Introducing $\lambda \triangleq \sum_{(ij)\in\mathcal{E}} \lambda_{ij}$ and $\mathcal{P}_\Lambda \triangleq \frac{2\Lambda}{\text{Tr }\Lambda}$, we write Λ as the product of these two more interpretable quantities to gain more insight on the following Laplacian:

$$\Lambda = \lambda \mathcal{P}_\Lambda. \tag{86}$$

In this setting, λ is the expected rate of communication over the *whole graph*, while $\mathcal{P}_\Lambda$ can be interpreted as the Laplacian of $\mathcal{E}$ with each edge weighted with its probability of having spiked given a communication happened in the graph. We have:

$$\chi_1[\Lambda] = \frac{\chi_1[\mathcal{P}_\Lambda]}{\lambda} \quad ; \quad \chi_2[\Lambda] = \frac{\chi_2[\mathcal{P}_\Lambda]}{\lambda}. \tag{87}$$

$\mathcal{P}_\Lambda$ being normalized, we could say that the quantities $\chi_1[\mathcal{P}_\Lambda], \chi_2[\mathcal{P}_\Lambda]$ characterize the graph's connectivity while λ is the global rate of communication. Then, using (87), since the following product is invariant by multiplication, we have:

$$2\sqrt{\chi_1[\mathcal{P}_\Lambda]\chi_2[\mathcal{P}_\Lambda]} = \sqrt{\chi_1[\Lambda]\chi_2[\Lambda]}\,\text{Tr}\,\Lambda, \tag{88}$$

meaning that the global communication rate is related the spectral quantity quantifying the graph's connectivity. If structural constraints on the network makes it impossible to verify communication rates are large enough, as the notion of time is *only* defined through the rate of gradient steps given by Assumption 3.1 (λ can be interpreted as *"the expected number of communications in the graph between the expected duration separating two subsequent gradient steps on a given node"*), it only means that the gradient steps are happening too fast compared to the ability of the network to communicate, and the rate of gradient steps should be slow-down by a factor of $\sqrt{2\chi_1\chi_2}$. We also note that the set of constraints ($\mathcal{C}_1$), ($\mathcal{C}_2$) or ($\mathcal{C}_3$) on Λ proposed in Section 4, and in particular the global budget on $\sum_{(ij)\in\mathcal{E}}\lambda_{ij}$, can be interpreted as a coarse model for physical or energetic limitations on total communication activity within a networked system. While this constraint does not necessarily lead to sparse or edge-dominated patterns (as in uniform or symmetric graphs), it enables us to explore how global communication limits shape optimal connectivity under fixed resources.

Appendix H. Practical Implementation

In this section, we describe in more detail the implementation of our algorithm. As we did not physically execute our method on a compute network but carried it out on a single machine, all the asynchronous computations and communications had to be simulated. Thus, we will first discuss the method we followed to simulate our asynchronous framework before detailing the practical steps of our algorithm through a pseudo-code.

H.1 Simulating the Poisson Point Processes

To emulate the asynchronous setting, before running our algorithm, we generate 2 independent sequences of jump times at the graph's scale: one for the computations and one for the communications. As we considered independent P.P.Ps, the time increments follow a Poisson distribution. At the graph's scale, each node spiking at a rate of 1, the Poisson parameter for the gradient steps process is n. Following the experimental setting of the Continuized framework (Even et al., 2021a), we considered that all edges in $\mathcal{E}$ had the same probability of spiking. Thus, given any graph $\mathcal{E}$ and $\mathcal{L}$ its corresponding standard Laplacian with unit edge weights, we computed the parameter λ of the communication process:

$$\lambda = \sqrt{2\chi_1\left[\frac{\mathcal{L}}{|\mathcal{E}|}\right]\chi_2\left[\frac{\mathcal{L}}{|\mathcal{E}|}\right]}. \tag{89}$$

Having generated the 2 sequences of spiking times at the graph's scale, we run our algorithm playing the events in order of appearance, attributing the *location* of the events by sampling uniformly one node if the event is a gradient step or sampling uniformly an edge in $\mathcal{E}$ if it is a communication.

H.2 Pseudo Code

We keep the notations introduced in Eq. (3) with values specified in Appendix C.1 and for the sake of completeness, we also specify the matrix $\mathcal{A}$ describing the linear ODE (8):

$$\mathcal{A} = \begin{pmatrix} -\eta & \eta & 0 & 0 & 0 & 0 \\ \tilde{\eta} & -\tilde{\eta} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\alpha & \alpha & 0 & 0 \\ 0 & -\theta\nu & -\theta & 0 & -\theta & 0 \\ 0 & 0 & 0 & 0 & -\alpha & \alpha \\ 0 & 0 & 0 & 0 & \tilde{\alpha} & -\tilde{\alpha} \end{pmatrix} \tag{90}$$

As described in Appendix H.1, we call `PPPspikes` the process mentioned above, returning the ordered sequence of events and time of spikes of the two P.P.Ps. Then, we can write the pseudo-code of our implementation of the DADAO optimizer in Algorithm 2.

Algorithm 2: Pseudo-code of our implementation of DADAO on a single machine.

```
Input: On each machine $i \in \{1, ..., n\}$, an oracle able to evaluate $\nabla f_i$, Parameters
       $\mu, L, \chi_1, t_{\max}, n, \lambda$.
       The graph $\mathcal{E}$.
1  Initialize on each machine $i \in \{1, ..., n\}$:
2      Set $X^{(i)} = (x_i, \tilde{x}_i, \tilde{y}_i)$ and $Y^{(i)} = (y_i, z_i, \tilde{z}_i)$ to 0 ;
3      Set constants $\nu, \tilde{\eta}, \eta, \gamma, \alpha, \tilde{\alpha}, \theta, \delta, \tilde{\delta}, \beta, \tilde{\beta}$ using $\mu, L, \chi_1$;
4      Set $\mathcal{A}$;
5      $T^{(i)} \leftarrow 0$ ;
6  ListEvents, ListTimes $\leftarrow$ PPPspikes$(n, \lambda, t_{\max})$ ;
7  $n_{\text{events}} \leftarrow |$ListEvents$|$ ;
8  for $k \in [\![1, n_{\text{events}}]\!]$ do
9      if ListEvents$[k]$ is to take a gradient step then
10         $i \sim \mathcal{U}([\![1, n]\!])$ ;
11         $\begin{pmatrix} X^{(i)} \\ Y^{(i)} \end{pmatrix} \leftarrow \exp\big((\texttt{ListTimes}[k] - T^{(i)})\mathcal{A}\big) \begin{pmatrix} X^{(i)} \\ Y^{(i)} \end{pmatrix}$ ;
12         $g_i \leftarrow (\nabla f_i(x_i) - \nu x_i - \tilde{y}_i)$;                 // Local gradient computation.
13         $x_i \leftarrow x_i - \gamma g_i$ ;
14         $\tilde{x}_i \leftarrow \tilde{x}_i - \tilde{\gamma} g_i$ ;
15         $\tilde{y}_i \leftarrow \tilde{y}_i + (\delta + \tilde{\delta}) g_i$ ;
16         $T^{(i)} \leftarrow$ ListTimes$[k]$ ;
17     else if ListEvents$[k]$ is to take a communication step then
18         $(i, j) \sim \mathcal{U}(\mathcal{E})$ ;
19         $\begin{pmatrix} X^{(i)} \\ Y^{(i)} \end{pmatrix} \leftarrow \exp\big((\texttt{ListTimes}[k] - T^{(i)})\mathcal{A}\big) \begin{pmatrix} X^{(i)} \\ Y^{(i)} \end{pmatrix}$ ;
20         $\begin{pmatrix} X^{(j)} \\ Y^{(j)} \end{pmatrix} \leftarrow \exp\big((\texttt{ListTimes}[k] - T^{(j)})\mathcal{A}\big) \begin{pmatrix} X^{(j)} \\ Y^{(j)} \end{pmatrix}$ ;
21         $m_{ij} \leftarrow (y_i + z_i - y_j - z_j)$;                           // Message exchanged.
22         $z_i \leftarrow z_i - \beta m_{ij}$;
23         $\tilde{z}_i \leftarrow \tilde{z}_i - \tilde{\beta} m_{ij}$;
24         $z_j \leftarrow z_j + \beta m_{ij}$;
25         $\tilde{z}_j \leftarrow \tilde{z}_j + \tilde{\beta} m_{ij}$;
26         $T^{(i)} \leftarrow$ ListTimes$[k]$;
27         $T^{(j)} \leftarrow$ ListTimes$[k]$;
28 return $(x_i)_{1 \le i \le n}$, the estimate of $x^*$ on each worker $i$.
```