Journal of Machine Learning Research 27 (2026) 1-57 Submitted 1/25; Revised 11/25; Published 2/26

DCatalyst: A Unified Accelerated Framework for Decentralized Optimization

Tianyu Cao[1] CAO357@PURDUE.EDU
Xiaokai Chen[1] CHEN4373@PURDUE.EDU
Gesualdo Scutari[1,2] GSCUTARI@PURDUE.EDU
[1] *Edwardson School of Industrial Engineering*
[2] *Elmore Family School of Electrical and Computer Engineering*
Purdue University, IN, USA.

Editor: Lin Xiao

Abstract

We study decentralized optimization over a network of agents, modeled as an undirected graph and operating without a central server. The objective is to minimize a composite function $f + r$, where f is a (strongly) convex function representing the average of the agents' losses, and r is a convex, extended-value function (regularizer).

We introduce *DCatalyst*, a unified black-box framework that injects Nesterov-type acceleration into decentralized optimization algorithms. At its core, DCatalyst is an *inexact*, *momentum-accelerated* proximal scheme (outer loop) that seamlessly wraps around a given decentralized method (inner loop). We show that DCatalyst attains optimal (up to logarithmic factors) communication and computational complexity across a broad family of decentralized algorithms and problem instances. In particular, it delivers accelerated rates for problem classes that previously lacked accelerated decentralized methods, thereby broadening the effectiveness of decentralized methods.

On the technical side, our framework introduces *inexact estimating sequences*–an extension of Nesterov's classical estimating sequences, tailored to decentralized, composite optimization. This construction systematically accommodates consensus errors and inexact solutions of local subproblems, addressing challenges that existing estimating-sequence-based analyses cannot handle while retaining a black-box, plug-and-play character.

Keywords: Acceleration, Decentralized optimization, Linear convergence, Variance reduction methods.

1. Introduction

We explore decentralized optimization across a network comprising $m > 1$ agents. The network is modeled as an undirected, static graph, possibly with no centralized nodes (e.g., servers). The agents aim to cooperatively solve the following optimization problem:

$$\min_{x\in\mathbb{R}^d} \; u(x) := f(x) + r(x), \quad \text{with} \quad f(x) := \frac{1}{m}\sum_{i=1}^{m} f_i(x). \tag{P}$$

Here $f_i : \mathbb{R}^d \to \mathbb{R}$ is the cost function associated with agent i, assumed to be smooth and (strongly) convex, and known only to that agent. Additionally, $r : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ is an

extended-value, convex function known to all agents, instrumental to enforce constraints or promote some desirable structure on the solution, such as sparsity or low rank.

The problem formulation (P) is notably versatile, encompassing a variety of applications across diverse fields. This paper places particular emphasis on (supervised) decentralized machine learning problems, specifically framed as Empirical Risk Minimization (ERM). In such formulations, each f_i has a finite-sum structure, resulting in the average of a predefined loss $\ell(\bullet;\xi):\mathbb{R}^d\to\mathbb{R}$ over data samples $\xi\in\mathcal{D}$, these being distributed among the agents in the network. Specifically, denoting by $\{\xi_{i,1},\cdots,\xi_{i,n}\}\subset\mathcal{D}$ the set of n samples accessed by agent i, drawn from an unknown distribution $\mathbb{P}$, the empirical risk f_i in (P) reads

$$f_i(x) := \frac{1}{n}\sum_{j=1}^{n}\underbrace{\ell(x;\xi_{i,j})}_{:=f_{ij(x)}}, \tag{1}$$

where $\ell(x;\xi_{i,j})$ quantifies the discrepancy between the hypothesized model, parameterized by x, and the data linked to $\xi_{i,j}$. Under mild assumptions on the loss function ℓ and i.i.d data distribution across the nodes, agents' cost functions f_i in (1) reflect statistical similarities in the data residing at different nodes, quantified by the following property

$$\big\|\big(\nabla f(x)-\nabla f_i(x)\big)-\big(\nabla f(y)-\nabla f_i(y)\big)\big\|\le\beta\|x-y\|,\quad\forall x,y\in\mathtt{dom}\,r, \tag{2}$$

which holds with high probability (Zhang and Lin, 2015; Hendrikx et al., 2020b). Here, $\beta=\widetilde{\mathcal{O}}(1/\sqrt{n})$ measures similarity of f_i's ($\widetilde{\mathcal{O}}$ hides log-factors and dependence on d).

In the decentralized setting, where no central node aggregates information, solving (P) requires decentralized algorithms that iteratively combine local computations with (possibly multiple) inter-node communication rounds (the latter being instrumental to acquire global information and inform the next computational tasks). Communication is typically the main bottleneck, often outweighing local computational costs (e.g., (Bekkerman et al., 2011; Lian et al., 2017)), which has motivated recent efforts toward *communication-efficient* methods.

Nesterov-type acceleration offers a natural route to reduce the number of iterations–and thus communication rounds–needed for convergence. While acceleration has been extensively developed in centralized and server-client settings (Nesterov, 1983; Beck and Teboulle, 2009; d'Aspremont et al., 2021), its decentralized counterparts remain unsatisfactory. Deferring to Sec. 1.1 a comprehensive review of the pertinent literature, it is sufficient to highlight here that existing accelerated decentralized methods are largely bespoke (see Tables 1-3): **(i)** they target *special cases* of (P) and often lack guarantees for broader classes such as *composite* problems ($r\neq 0$), which are central in machine learning; **(ii)** they typically do not exploit structural properties like similarity (2) that could improve communication efficiency, especially under ill-conditioning; and **(iii)** their convergence analyses do not generalize to new algorithmic families beyond the specific schemes and problem-classes considered.

This reveals a clear gap: a *universal* analytical framework capturing a broad class of accelerated decentralized algorithms is missing. This paper addresses precisely this gap by proposing a *unified black-box* acceleration framework *à la Catalyst* (Güler, 1992; Lin et al., 2015, 2018; d'Aspremont et al., 2021). Our approach, termed *Decentralized Catalyst* (DCatalyst), brings Nesterov-style acceleration to a wide family of existing decentralized algorithms that were not originally designed to be accelerated, making them applicable

to Problem (P) in its full generality. Notably, DCatalyst attains *optimal* communication complexity (up to logarithmic factors) for most of these methods, while offering substantial design flexibility: it can explicitly exploit functional structures (such as similarity and nonsmooth composite regularization) and allows one to trade off communication overhead against local computational cost.

1.1 Related Works

Given the focus on decentralized algorithms, our literature review is restricted to accelerated methods designed for mesh networks, and does not cover the extensive work on centralized (including server–client) systems. We group relevant decentralized algorithms into three classes, according to the specific instances of Problem (P) they address.

(i) Strongly convex f (Table 1): There exists a large body of accelerated decentralized first-order methods for this case, mostly for smooth, unconstrained problems (hence $r = 0$). These schemes achieve linear convergence in terms of communication complexity, but with different dependences on function and network parameters. Early methods leveraging a primal-dual reformulation of (P) (with $r = 0$) or invoking penalty strategies (Scaman et al., 2017; Kovalev et al., 2020; Li and Lin, 2020; Rogozin et al., 2021; Dvinskikh and Gasnikov, 2021; Kovalev et al., 2021; Li et al., 2020b; Uribe et al., 2020), attain rates scaling as $\mathcal{O}(\sqrt{\kappa_\ell}/\sqrt{1-\rho})$, where $\kappa_\ell := L_{\max}/\mu_{\min}$ is the *local* condition number ($L_{\max}$ and $\mu_{\min}$ are the largest smoothness and smallest strong convexity constants across the f_i's), and $\rho \in [0, 1)$ quantifies network connectivity (see Sec. 4 for the definition). This dependence on κ_ℓ is undesirable, since κ_ℓ may be much larger than the *global* condition number $\kappa_g := L/\mu$ of the aggregate loss f. More recent works (Ye et al., 2020, 2023; Song et al., 2024) improve the dependence to $\mathcal{O}(\sqrt{\kappa_g}/\sqrt{1-\rho})$, with (Ye et al., 2020, 2023) also covering composite problems ($r \neq 0$).

These methods are first-order and do not exploit possible *function similarity*, e.g., of the form (2). This limitation is particularly critical for ill-conditioned problems, where κ_g is very large. In many ERM models, the regularization parameter that optimizes test performance is tiny; for instance, in ridge regression one typically has $\mu = \mathcal{O}(1/\sqrt{mn})$, $L = \mathcal{O}(1)$, and $\beta = \mathcal{O}(1/\sqrt{n})$ (Zhang and Lin, 2015), so that $\kappa_g = \mathcal{O}(\sqrt{mn})$ while $\beta/\mu = \mathcal{O}(\sqrt{m})$. The latter can be much smaller and does not grow with the local sample size n. This has motivated a line of *centralized* work on exploiting similarity via statistical preconditioning (Shamir et al., 2014; Reddi et al., 2016; Yuan and Li, 2020; Zhang and Lin, 2015; Fan et al., 2023), sometimes combined with acceleration (Hendrikx et al., 2020b; Dvurechensky et al., 2022). The state-of-the-art method (Hendrikx et al., 2020b) achieves linear convergence with the number of communication rounds scaling (asymptotically) like $1 + \mathcal{O}(\sqrt{\beta/\mu})$, which is significantly better than $\sqrt{\kappa_g}$ (albeit at increased computation costs compared to first-order methods) and matches known lower bounds up to logarithmic factors (Arjevani and Shamir, 2015). However, these schemes are intrinsically *centralized*: they rely on a server node directly connected to all clients and are therefore infeasible on mesh networks.

To our knowledge, Network-DANE (Li et al., 2020a) and SONATA (Sun et al., 2022) are the only methods that exploit statistical similarity over mesh networks, but they do not incorporate acceleration, leading to communication complexity $\widetilde{\mathcal{O}}\big((1/\sqrt{1-\rho})\cdot(\beta/\mu)\cdot\log(1/\epsilon)\big)$ instead of the desired $\mathcal{O}(\sqrt{\beta/\mu})$ scaling. At present, there is no theoretical machinery that

Table 1: **Decentralized accelerated algorithms for strongly convex f in** (P): The most efficient decentralized accelerated algorithms in the literature for different instances of (P). Key parameters include (definitions in Sec. 2): $\kappa_g := L/\mu$ (global condition number of f), $\kappa_\ell := L_{\max}/\mu_{\min}$ (local condition number of f_i's), β (function similarity, (2)), and $\rho \in [0,1)$ (network connectivity). Notation $\widetilde{\mathcal{O}}$ omits poly-log factors of problem parameters, independent of ϵ. Notably, the proposed methods stands as the first to achieve acceleration either in terms of gradient or communication complexity, with the latter harnessing similarity ((Tian et al., 2022) is conference version of this paper)

Algorithm	**Problem**	**# (Proximal) gradient**	**# Communications**
OPAPC (Kovalev et al., 2020)	f_i scvx, $r \equiv 0$	$\mathcal{O}\left(\sqrt{\kappa_\ell}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\sqrt{\frac{\kappa_\ell}{1-\rho}}\log\frac{1}{\epsilon}\right)$
Mudag (Ye et al., 2023)		$\mathcal{O}\left(\sqrt{\kappa_g}\log\frac{1}{\epsilon}\right)$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_g}{1-\rho}}\log\frac{1}{\epsilon}\right)$
DPAG (Ye et al., 2020)	f scvx, $r \neq 0$	$\mathcal{O}\left(\sqrt{\kappa_g}\log\frac{1}{\epsilon}\right)$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_g}{1-\rho}}\log\frac{1}{\epsilon}\right)$
Proposed: DCatalyst-SONATA-L		$\widetilde{\mathcal{O}}\left(\sqrt{\kappa_g}\log\frac{1}{\epsilon}\right)$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_g}{1-\rho}}\log\frac{1}{\epsilon}\right)$
Proposed: DCatalyst-SONATA-F		$\widetilde{\mathcal{O}}\left(\sqrt{\frac{L+\beta}{\beta}}\cdot\frac{\beta}{\mu}\log^2\frac{1}{\epsilon}\right)$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\beta/\mu}{1-\rho}}\log\frac{1}{\epsilon}\right)$

.

accelerates these methods while simultaneously harnessing the benefits of function similarity. The framework proposed in this paper fills precisely this gap (see Sec. 1.2).

(ii) (non-strongly) convex f (Table 2): The literature on accelerated decentralized methods for (non-strongly) convex instances of (P) is relatively sparse. Existing schemes (Li et al., 2020b; Li and Lin, 2020; Uribe et al., 2020; Dvinskikh and Gasnikov, 2021) focus on smooth objectives ($r = 0$). Among these, (Li et al., 2020b) is notable for achieving accelerated rates in terms of both gradient and communication oracles, namely $\mathcal{O}(\sqrt{L_{\max}/\epsilon})$ and $\mathcal{O}(\sqrt{L_{\max}/((1-\rho)\cdot\epsilon)})$, although the dependence is on $L_{\max}$ rather than the more favorable global constant L. For composite problems ($r \neq 0$), the landscape is even more limited: the only decentralized method in this category, (Li et al., 2020b), does *not* reach full acceleration, with gradient and communication complexities $\mathcal{O}(L_{\max}/\epsilon)$ and $\mathcal{O}(L_{\max}/(\sqrt{1-\rho}\cdot\epsilon))$, instead of the expected $\mathcal{O}(\sqrt{L/\epsilon})$ and $\mathcal{O}(\sqrt{L/((1-\rho)\cdot\epsilon)})$. Moreover, none of these works exploit potential function similarity. Our framework fills these gaps by providing optimal acceleration (up to log-factors) for composite problems (e.g., when applied to SONATA (Sun et al., 2022)) and by allowing similarity to be explicitly leveraged whenever present.

(iii) f_i's with finite-sum structure (Table 3): In the *centralized* setting, the combination of variance reduction (VR) and acceleration is well understood: numerous methods exploit the finite-sum structure of the f_i's while retaining fast rates (Lin et al., 2015; Allen-Zhu, 2018; Zhou et al., 2018, 2019; Lan et al., 2019; Driggs et al., 2022). By sampling component gradients, VR schemes enable cheaper stochastic steps without sacrificing convergence speed. In contrast, *decentralized* accelerated methods with VR are much rarer, and currently restricted to *smooth* objectives ($r = 0$) (Hendrikx et al., 2020a, 2021; Li et al., 2022). To the best of our knowledge, the only VR-based decentralized method for composite problems is (Ye et al., 2026), which, however, does not provide any acceleration. Our framework offers the first systematic design of decentralized VR-based algorithms for composite objectives that provably achieve accelerated rates, thereby closing this gap in the current decentralized optimization literature.

Table 2: **Decentralized accelerated algorithms for (nonstrongly) convex f**: The most efficient decentralized accelerated algorithm for each reported instance of (P), based either on gradient or communication complexity. Definitions are as in Table 1. For nonsmooth u, ϵ-optimality is measured in terms of the gradient of the Moreau envelope of u. The proposed framework enables acceleration for nonsmooth u as well as rate dependence on L (rather than the less favorable $L_{\max}$).

Algorithm	Problem	# (Proximal) gradient	# Communications
APM-C (Li et al., 2020b)	f cvx, $r \equiv 0$	$\mathcal{O}\left(\sqrt{\frac{L_{\max}}{\epsilon}}\right)$	$\mathcal{O}\left(\sqrt{\frac{L_{\max}}{(1-\rho)\epsilon}}\log\frac{1}{\epsilon}\right)$
APM (Li et al., 2020b)	f cvx, $r \neq 0$	$\mathcal{O}\left(\frac{L_{\max}}{\epsilon}\right)$	$\mathcal{O}\left(\frac{L_{\max}}{\epsilon\sqrt{1-\rho}}\right)$
Proposed: DCatalyst-SONATA-L		$\widetilde{\mathcal{O}}\left(\sqrt{\frac{L}{\epsilon}}\log\frac{1}{\epsilon}\right)$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{L}{(1-\rho)\epsilon}}\log\frac{1}{\epsilon}\right)$
Proposed: DCatalyst-SONATA-F		$\widetilde{\mathcal{O}}\left(\sqrt{\frac{L+\beta}{\epsilon}}\log^2\frac{1}{\epsilon}\right)$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\beta}{(1-\rho)\epsilon}}\log\frac{1}{\epsilon}\right)$

Table 3: **Decentralized accelerated algorithms for finite-sum f_i's**: Current decentralized accelerated VR algorithms for (P) with f_i's having a finite-sum structure. Same quantities as defined in Table 1; in addition, $\tilde{\kappa}_s := (\max_{ij} L_{ij})/\mu_{\min}$, $\kappa_s := (\max_i (1/n)\sum_{j=1}^n L_{ij})/\mu_{\min}$, and $b \in \{1, \cdots, n\}$ represents the batch-size used in the (stochastic) estimation of each ∇f_i. Here, L_{ij} is smoothness parameter of f_{ij}. Notice that $\kappa_\ell \leq \kappa_s \leq n\kappa_\ell$ and $\kappa_s \leq \tilde{\kappa}_s \leq mn\kappa_s$ (details in Sec. 2). The proposed framework achieves acceleration also when u is nonsmooth.

Algorithm	Problem	# (Proximal) Gradient	# Communications
ADFS(Hendrikx et al., 2021)	f_i scvx, $r \equiv 0$	NA	$\mathcal{O}\left(\sqrt{\frac{\kappa_\ell}{1-\rho}}\log\frac{1}{\epsilon}\right)$
Acc-VR-EXTRA-CA (Li et al., 2022)		$\mathcal{O}\left(\left(\sqrt{n\kappa_s}+n\right)\log\frac{1}{\epsilon}\right)$	
DVR-Catalyst (Hendrikx et al., 2020a)		$\widetilde{\mathcal{O}}\left(\left(\sqrt{n\kappa_s}+n\right)\log\frac{1}{\epsilon}\right)$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_\ell}{1-\rho}}\sqrt{\frac{n\kappa_\ell}{\kappa_s}}\log\frac{1}{\epsilon}\right)$
Proposed: DCatalyst-VR-EXTRA		$\widetilde{\mathcal{O}}\left(\sqrt{\kappa_\ell n b}\log\frac{1}{\epsilon}\right)$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_\ell}{1-\rho}}\sqrt{\frac{n}{b}}\log\frac{1}{\epsilon}\right)$
PMGT-LSVRG (Ye et al., 2026)	f scvx, $r \neq 0$	$\mathcal{O}\left(\left(n+\tilde{\kappa}_s\right)\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{\tilde{\kappa}_s\log\tilde{\kappa}_s+n\log n}{\sqrt{1-\rho}}\log\frac{1}{\epsilon}\right)$
Proposed: DCatalyst-PMGT-LSVRG		$\widetilde{\mathcal{O}}\left(\sqrt{\kappa_s n}\log\frac{1}{\epsilon}\right)$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_s n}{1-\rho}}\log\frac{1}{\epsilon}\right)$

1.2 Main Contributions

Our technical contributions can be summarized as follows.

(i) DCatalyst–A unified decentralized acceleration framework: We propose a unified black-box framework, *Decentralized Catalyst* (DCatalyst), that injects Nesterov-type acceleration into *decentralized* optimization algorithms, thereby broadening their applicability to effectively tackle the full spectrum of Problem (P). DCatalyst is compatible with the majority of decentralized optimization algorithms–specifically capable of linear convergence in solving strongly convex instances of (P). At its core, DCatalyst is an *inexact*, *momentum-accelerated* proximal scheme (forming the outer loop) that wraps around the selected decentralized (non accelerated) algorithm (inner loop), which approximately solves the auxiliary proximal subproblems associated with (P) according to a newly designed *inexactness criterion* and *warm-start policy* tailored to decentralized settings.

We prove that DCatalyst attains optimal (up to logarithmic factors) communication and/or computational complexity across a wide range of embedded decentralized algorithms

and problem instances. Numerical experiments support our theory, demonstrating consistent acceleration over the corresponding non-accelerated baselines, with particularly pronounced gains on ill-conditioned problems. DCatalyst also outperforms existing *single-loop* decentralized accelerated methods that are tailored to specific subclasses of (P), highlighting its robustness and versatility.

(ii) New decentralized accelerated algorithms for unsolved instances of (P)**:** The double-loop structure of DCatalyst is key to enabling acceleration while preserving the key structural features of the chosen inner-loop algorithm. This yields several *new* decentralized accelerated methods for problem instances that were previously out of reach. Notable examples include:

- *Leveraging function similarity:* For strongly convex objectives u (possibly with $r \neq 0$) satisfying the similarity condition (2), combining DCatalyst with SONATA-F (Sun et al., 2022)–which already exploits similarity via local preconditioning–yields, for the first time, nearly optimal (accelerated) communication complexity (Table 1), $\widetilde{\mathcal{O}}((1/\sqrt{1-\rho})\cdot\sqrt{\beta/\mu}\cdot\log(1/\epsilon))$, improving upon the non-accelerated $\mathcal{O}((\beta/\mu)/\sqrt{1-\rho}\cdot\log(1/\epsilon))$ scaling of SONATA-F.

- *Acceleration for nonstrongly convex, composite objectives:* DCatalyst provides accelerated schemes for merely convex composite functions u, possibly exploiting similarity when present. In particular, it adeptly extends decentralized methods that are only known to converge linearly in the strongly convex case into accelerated algorithms for the convex case. This gives rise to new algorithms such as DCatalyst-SONATA-L and DCatalyst-SONATA-F (Table 2).

- *Acceleration for composite u with finite-sum structure f_i's:* DCatalyst offers a natural platform to integrate variance-reduction (VR) techniques within decentralized schemes for composite objectives with finite-sum local losses. For instance, building on the PMGT-LSVRG algorithm (Ye et al., 2026), we design DCatalyst-PMGT-LSVRG, which achieves, for the first time, accelerated convergence in both computation and communication. This is especially beneficial for tackling ill-conditioned problems, where the number of data points n is significantly smaller than the stochastic condition number κ_s (cf. Table 3).

(iii) New analysis: the notion of *inexact* estimating sequences: A key technical contribution concerns the analytical framework. Directly applying the classical, *centralized* Catalyst framework and its variants (Güler, 1992; Lin et al., 2015, 2018; d'Aspremont et al., 2021) as a black-box accelerator in decentralized settings is not viable, for the following reasons. **(a)** *Inner subproblems and termination:* Catalyst is built around a *single* sequence of decision variables, whereas decentralized methods maintain multiple agent-specific iterates and auxiliary variables (e.g., gradient-tracking or dual variables). It is therefore unclear how to define inner subproblems compatible with the centralized Catalyst formulation or on which sequence and metric its function-value–based termination rule should act. A naive remedy is to apply Catalyst's subproblem to a surrogate sequence mimicking a centralized iterate (e.g., the average of the agents' iterates) and enforce termination on the objective-value gap evaluated on this iterate. However, convergence of decentralized algorithms is typically certified via Lyapunov functions that mix iterate gaps, consensus errors,

and auxiliary states; reconciling these Lyapunov-based metrics with Catalyst's function-gap criterion forces *ad-hoc, algorithm-specific* analyses tailored to the chosen surrogate and Lyapunov structure, thereby *destroying the black-box, problem-agnostic nature and universality of Catalyst*. This is precisely what happens in Catalyst-EXTRA (Li and Lin, 2020) and DVR-Catalyst (Hendrikx et al., 2020a), which adapt centralized Catalyst to EXTRA and DVR via problem- and algorithm-specific inner termination rules; these rules and analyses do not extend, for instance, to *composite* objectives ($r \neq 0$). **(b)** *Inner-loop warm restart:* The centralized Catalyst framework also provides no guidance on how to initialize a decentralized algorithm at the beginning of each inner loop, in particular its auxiliary variables. If one simply restarts the inner method as if it were run in isolation, the Lyapunov function at the start of a new inner loop may be much larger than at the end of the previous one, potentially breaking outer-loop convergence or nullifying acceleration. A *careful warm-start* strategy that exploits outer-loop convergence–especially for the auxiliary variables–is therefore essential, but it is not addressed in the Catalyst framework.

To systematically address the issues above, we introduce the *inexact estimating sequences* framework, a novel extension of classical estimating sequences (Nesterov, 2013) tailored to composite decentralized optimization. Our construction modifies the estimating-sequence definition to explicitly incorporate *exogenous* error sequences, thereby **(i)** handling multiple agent-specific variable sequences and rigorously capturing both consensus errors and inexact decentralized subproblem solutions—features that classical constructions (Nesterov, 2013; Baes, 2009; Lin et al., 2015, 2018; d'Aspremont et al., 2021) cannot capture while preserving a genuine black-box character; and **(ii)** providing a unified framework for convergence analysis and parameter tuning of decentralized algorithms, provably yielding acceleration. Guided by this framework, **(iii)** we further derive new inner-loop termination criteria and warm-start policies for the adopted decentralized algorithms that guarantee overall accelerated convergence. The resulting stopping rules are *weaker* than those in Catalyst, naturally accommodate *composite* losses, and allow generic (possibly stochastic) decentralized algorithms in the inner loop while still guaranteeing accelerated rates. Compared with the original Catalyst framework (Güler, 1992; Lin et al., 2015, 2018; d'Aspremont et al., 2021) and its decentralized instantiations (Li and Lin, 2020; Hendrikx et al., 2020a), our approach restores a true black-box, plug-and-play acceleration mechanism aligned with the structural and analytical specificities of decentralized (composite) optimization.

1.3 Notation and paper structure

We use following notations. Let $[m] := \{1, \cdots, m\}$, with m being any positive integer. We use $\|.\|_p$ to denote the ℓ_p norm; $\|.\|$ is by default the ℓ_2-norm when applied to vectors and the Frobenius norm when we input matrices. We denote by 1_m the the m-dimensional (column) vector of all ones. Bold letters is used to denote matrices resulting from stacking a set of given vectors row-wise; for example, given $x_1, \ldots, x_m \in \mathbb{R}^d$, we define $\mathbf{x} := [x_1, \cdots, x_m]^\top \in \mathbb{R}^{m \times d}$ and $\bar{x} := (1/m)\sum_{i=1}^m x_i$. For a square matrix A, we use $\sigma_{\max}(A)$ and $\sigma^+_{\min}(A)$ respectively to denote its maximum eigenvalue and smallest positive eigenvalue. For a sequence of sets $\{B_k\}_{k=0}^\infty$, we use $\prod_{k=0}^\infty B_k$ to denote their Cartesian product.

Given $\eta > 0$ and a proper, closed convex function $r : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$, the Moreau envelope of r, $\mathtt{M}_{\eta r} : \mathbb{R}^d \to \mathbb{R}$, and the proximal operator, $\mathtt{prox}_{\eta r}(x) : \mathbb{R}^d \to \mathbb{R}^d$, are

$$\mathtt{M}_{\eta r}(x) := \min_{y\in\mathbb{R}^d} \eta\, r(y) + \frac{1}{2}\|y-x\|^2 \quad \text{and} \quad \mathtt{prox}_{\eta r}(x) := \arg\min_{y\in\mathbb{R}^d} \eta\, r(z) + \frac{1}{2}\|y-x\|^2,$$

respectively. Notice that $\mathtt{M}_{\eta r}$ is differentiable, with gradient

$$\nabla \mathtt{M}_{\eta r}(x) = \frac{1}{\eta}\left(x - \mathtt{prox}_{\eta r}(x)\right),$$

which is thus $1/\eta$-Lipschitz continuous. Denote by $\mathtt{M}^\star_{\eta r}$ the minimum of $\mathtt{M}_{\eta r}$ over $\mathtt{dom}\, r$. With a slight abuse of notation, given $\mathbf{x} \in \mathbb{R}^{m\times d}$, we will write $\mathbf{y} := \mathtt{prox}_{\eta r}(\mathbf{x}) \in \mathbb{R}^{m\times d}$, to denote in short row-wise equalities, that is, $y_i = \mathtt{prox}_{\eta r}(x_i)$, for all $i \in [m]$.

The rest of the paper is organized as follows. In Sec. 2, we introduce the main assumptions underlying Problem (P); Sec. 3 formally introduces the proposed framework, decentralized Catalyst, along with its convergence properties for strongly convex and convex losses. Sec. 4 applies the proposed framework to a variety of decentralized algorithms. Sec. 6 provides the proofs of the main results. Finally, some numerical results validating the effectiveness of our framework are discussed in Sec. 7. A summary of the symbols and notations used in the paper is reported in Appendix. C.

2. Assumptions and Preliminaries

We study (P) over a mesh network. The network is modeled as a static undirected graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} := [m]$ denotes the set of agents and $\mathcal{E} := \{(i,j)|i,j \in \mathcal{V}\}$ is the set of edges: agents i and j can communicate if $(i,j) \in \mathcal{E}$. We make the blanket assumption that $\mathcal{G}$ is connected, and $(i,i) \in \mathcal{E}$ for all $i \in [m]$. Following are assumptions on Problem (P).

Assumption 1 *For Problem* (P)*, the following hold:*

(i) Each $f_i : \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable, μ_i-strongly convex and L_i-smooth, with $0 \le \mu_i \le L_i < \infty$;

(ii) $f : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex and L-smooth, with $0 \le \mu \le L < \infty$;

(iii) $r : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ is a proper, closed, and convex function.

This assumption encompasses both strongly convex or convex functions f, with the latter corresponding to $\mu = 0$. Associated to Assumption 1, there are the following quantities:

$$L_{\max} := \max_{i\in[m]} L_i, \quad \mu_{\min} := \min_{i\in[m]} \mu_i, \quad \kappa_g := \frac{L}{\mu}, \quad \kappa_\ell := \frac{L_{\max}}{\mu_{\min}}.$$

Here, by writing κ_g and κ_ℓ we implicitly assume $\mu > 0$ and $\mu_{\min} > 0$, respectively. These quantities are pivotal in assessing the efficiency of decentralized algorithms. It is noteworthy that $\kappa_g \le \kappa_\ell$. The gap can be significantly large, as shown in the following example.

Example 1 *Let* $f_1, f_2 : \mathbb{R}^2 \to \mathbb{R}$ *such that*

$$f_1(x) = \frac{1}{2}x^\top \begin{pmatrix} 1 & 0 \\ 0 & \epsilon \end{pmatrix} x, \quad f_2(x) = \frac{1}{2}x^\top \begin{pmatrix} \epsilon & 0 \\ 0 & 1 \end{pmatrix} x.$$

Hence, $f(x) = ((1+\epsilon)/4)\|x\|^2$. *It follows that* $\kappa_g = 1$ *while* $\kappa_\ell = \epsilon^{-1}$, *implying that* κ_ℓ *can be arbitrarily large.* □

When dealing with ERM problems with f in the form (1), we further postulate the following standard property for the f_{ij}'s.

Assumption 2 *Each* $f_{ij} : \mathbb{R}^d \to \mathbb{R}$ *in (1) is continuously differentiable, convex, and* L_{ij}*-smooth, with* $0 < L_{ij} < \infty$.

In the context of Assumption 2, we define several critical quantities that are instrumental in the analysis of first-order methods solving in particular ERM problems:

$$\bar{L}_i := \frac{1}{n}\sum_{j=1}^n L_{ij}, \quad \bar{L}_{\max} := \max_{i\in[m]} \bar{L}_i, \quad \kappa_s = \frac{\bar{L}_{\max}}{\mu_{\min}}.$$

Since $L_i \le \bar{L}_i \le nL_i$, we have

$$\kappa_\ell \le \kappa_s \le n\kappa_\ell \quad \text{and} \quad L_{\max} \le \bar{L}_{\max} \le nL_{\max}. \tag{3}$$

Here, κ_s represents the stochastic condition number (in contrast to the "batch" condition numbers, κ_g and κ_ℓ), which appears in the convergence rates of VR-based algorithms for finite-sum optimization problems. The relationship (3) implies that, in the worst-case scenario where all f_{ij}'s are independent, VR-based methods do not offer any advantage over their batch counterparts. However, ERM problems featuring correlated data samples often exhibit $\kappa_s \ll n\kappa_\ell$, indicating a potential for significant improvements in the computational efficiency of VR-methods compared to batch algorithms.

In the exploration of ERM problems, beyond the settings outlined in Assumptions 1 and 2, we anticipated in (2) additional structure in loss functions f_i, termed function similarity. We encapsulate this property in the subsequent assumption for ease of reference.

Assumption 3 *Each* f_i *in Assumption 1,* $i \in [m]$, *satisfies:*

$$\big\|\big(\nabla f(x) - \nabla f_i(x)\big) - \big(\nabla f(y) - \nabla f_i(y)\big)\big\| \le \beta\|x - y\|, \quad \forall x, y \in \mathbf{dom} r,$$

for some $\beta > 0$. *It is assumed* $\beta > \mu$ *without loss of generality.*

The smaller β, the more similar f_i's. Under Assumption 1, the following bounds holds for β: $\beta \le \max_{i\in[m]} \max\big\{|L_i - \mu|, |\mu - L_i|\big\}$. The example below demonstrates the typical, more favorable scaling of β in ERM scenarios, due to statistical similarity.

Example: Consider the ERM setting (1), where data are i.i.d., locally and across the agents. Assume $\nabla^2 f_i$ is M-Lipschitz and under extra (mild) assumptions–see (Zhang and Xiao, 2018)–the following bounds hold for β, with high probability:

$$\beta = \widetilde{\mathcal{O}}\left(\sqrt{\frac{L^2}{n}}\right), \text{ if } M = 0 \ (f_i\text{'s are quadratic}); \quad \text{or} \quad \beta = \widetilde{\mathcal{O}}\left(\sqrt{\frac{L^2 d}{n}}\right), \text{ otherwise.}$$

It turns out that

$$\frac{\beta}{\mu}\Big/\frac{L}{\mu} = \min\left\{1, \widetilde{\mathcal{O}}\left(\sqrt{\frac{d}{n}}\right)\right\}.$$

Hence, the ratio β/μ may be considerably smaller than κ_g, particularly in cases where n is large, a common condition in many applications. This indicates potential for improved efficiency in exploiting function similarity within decentralized optimization.

3. The Decentralized Catalyst Framework

DCatalyst inputs a given decentralized algorithm and enhances it through a black-box acceleration procedure. Our initial step involves identifying a spectrum of candidate algorithms via a broad set of assumptions, which encapsulate the vast majority of current decentralized schemes–see Sec. 3.1. Sec. 3.2 proceeds to introduce the black-box procedure enabling acceleration in the aforementioned decentralized schemes. Convergence analysis of DCatalyst is presented in Sec. 3.3 (strongly convex u) and Sec. 3.4 (convex u).

3.1 Distributed algorithms

We model decentralized algorithms through operators that iteratively act on variables held by the agents. To this end, let us denote by $s_i = [x_i^\top, y_i^\top]^\top$ the set of variables controlled by agent i, where $x_i \in \mathbb{R}^d$ is the local estimate of the optimization variable x of the problem under consideration, and $y_i \in \mathbb{R}^{d'}$ serves as some auxiliary vector-variable, instrumental to track locally some additional information used by the algorithm to generate agents' updates, such as dual (e.g., (Alghunaim et al., 2020; Ling et al., 2015; Shi et al., 2015)) or gradient-tracking estimates (e.g., (Di Lorenzo and Scutari, 2016; Nedić et al., 2017; Qu and Li, 2017; Sun et al., 2022)). Denote by s_i^t the values of these variables at iteration t, and by $\boldsymbol{s}^t := (s_i^t)_{i\in[m]}$ the stack of the agents' tuples at that iteration. The decentralized algorithm under consideration is then modeled as a (possibly random) mapping $\mathcal{M} : \mathbb{R}^{m\times(d+d')} \to \mathbb{R}^{m\times(d+d')}$: given $\boldsymbol{s}^t$ as input at iteration t, the decentralized algorithm $\mathcal{M}$ produced $\boldsymbol{s}^{t+1} = \mathcal{M}(\boldsymbol{s}^t)$ as new iterate at time $t+1$.

We focus on accelerating decentralized algorithms $\mathcal{M}$ enjoying the following properties.

Assumption 4 *Let the decentralized algorithm $\mathcal{M}$ be employed to solve a strongly convex instance of Problem* (P) *(that is, under Assumption 1, with $\mu > 0$) over a mesh network. Let $\{\mathbf{s}^t\}_t$ be the sequence of state variables generated by $\mathcal{M}$, starting from a given, feasible $\mathbf{s}^0$. There exists a merit function $\mathcal{L} : \mathbf{s} \mapsto \mathcal{L}(\mathbf{s}) \in \mathbb{R}_+$ associated with $\mathcal{M}$, such that*

(i) (linear convergence):

$$\mathbb{E}[\mathcal{L}(\boldsymbol{s}^t)|\boldsymbol{s}^0] \le c_{\mathcal{L}}\left(1 - \frac{1}{r_{\mathcal{M}}}\right)^t \mathcal{L}(\boldsymbol{s}^0), \quad \forall t \ge 0, \tag{4}$$

where $c_{\mathcal{L}} > 0$ and $r_{\mathcal{M}} > 1$ are problem/algorithm-dependent constants; and

(ii) (optimality): Given any $\boldsymbol{s} = [\boldsymbol{x}^\top, \boldsymbol{y}^\top]^\top \in \mathbb{R}^{m\times(d+d')}$,

$$\frac{1}{m}\|\boldsymbol{x} - \boldsymbol{x}^\star\|^2 \le \mathcal{L}(\boldsymbol{s}),$$

where $\boldsymbol{x}^\star := 1_m (x^\star)^\top$ *and* $x^\star$ *is a solution of the optimization problem.*

Assumption 4 is quite standard in the literature of decentralized optimization algorithms– Sec. 4 specializes (i) and (ii) to the most common schemes. Specifically, (i) states that, when minimizing a strongly convex loss on a network, $\mathcal{M}$ exhibits a linear convergence rate. Condition (ii) guaranteed that convergence of the merit function $\mathcal{L}$ implies that of the iterates to the optimal solution $x^\star$ (at the same rate).

3.2 Accelerating $\mathcal{M}$ via decentralized Catalyst

The DCatalyst framework is formally introduced in Algorithm 1 and commented next. At its core, it can be viewed as the successive application of the chosen decentralized algorithm $\mathcal{M}$ for the inexact minimization of the function $u^k(x)$, defined in (5). The quadratic term $(\delta/2)\|x - z_i^k\|^2$ therein functions as an extrapolation mechanism akin to Nesterov's approach, to gain acceleration. The momentum variables z_i^k are updated locally by the respective agents at the conclusion of each outer loop (during the Extrapolation step (S.2)), following the update rule in (6). Crucially, at the start of every inner loop, the decentralized algorithm $\mathcal{M}$ undergoes a strategic *warm-restart*, and runs until a predefined *termination criterion* is met. These strategies are informed directly from our convergence analysis, with their specifics being dependent on both the optimization problem and the decentralized algorithm $\mathcal{M}$ at hand. Details on these criteria along with tuning recommendations are provided in Sec. 3.3 and Sec. 3.4 for strongly convex and non-strongly convex functions u, respectively.

A Bird's-eye view: DCatalyst can be viewed as an acceleration of a *centralized*, inexact proximal method (the outer loop), where the proximal subproblems are approximately solved in a distributed manner using the decentralized algorithm $\mathcal{M}$. Assuming one can absorb consensus errors on the agents' variables x_i's and momentum vectors z_i's into this criterion of solution approximation (to be introduced), we can approximate `(S.1)` and `(S.2)` as

$$
\begin{aligned}
& x_i^k \approx \bar{x}^k := \frac{1}{m}\sum_{i=1}^m x_i^k \quad \text{and} \quad z_i^k \approx \bar{z}^k := \frac{1}{m}\sum_{i=1}^m z_i^k, \\
& \texttt{(S.1)'}: \quad \bar{x}^{k+1} \approx \arg\min_{x\in\mathbb{R}^d} u(x) + \frac{\delta}{2}\|x - \bar{z}^k\|^2, \\
& \texttt{(S.2)'}: \quad \bar{z}^{k+1} = \bar{x}^{k+1} + \frac{\alpha^k(1-\alpha^k)}{(\alpha^k)^2 + \alpha^{k+1}}\,(\bar{x}^{k+1} - \bar{x}^k),
\end{aligned}
$$

where we used the fact that the minimization of $u^k(x)$ in (5) and that of the function in `(S.1)'` have the same solution. The above dynamics are a resemble of an inexact proximal acceleration (Lin et al., 2015, 2018; d'Aspremont et al., 2021).

Next we provide a novel criterion of inexactness for solving the subproblems (P^k) distributively along with a suitable warm-start policy of the inner-algorithm $\mathcal{M}$, yielding acceleration. Sec. 3.3 considers strongly convex objectives while Sec. 3.4 focuses on (merely) convex ones.

Algorithm 1 `Decentralized Catalyst`

Input: $x_i^0 = z_i^0 = z_i^{-1} \in \texttt{dom}\ r,\ \ i \in [m]$; $\delta > 0$;
$K \in \mathbb{N}_{++}$ (# of outer loops), $\{\alpha^k \in (0,1)\}_{k=0}^K$; Decentralized algorithm $\mathcal{M}$.

Output: $\mathbf{x}^K = (x_i^K)_{i\in[m]}$

for $k = 0, 1, 2, \ldots, K-1$ **do**

`Set`

$$u^k(x) = \frac{1}{m}\sum_{i=1}^m f_i^k(x) + r(x), \quad \text{with} \quad f_i^k(x) = f_i(x) + \frac{\delta}{2}\|x - z_i^k\|^2; \tag{5}$$

`(S.1) Minimization step:` Minimize approximately u^k over the network $\mathcal{G}$ running $\mathcal{M}$, with proper initialization $s_i^k = [(x_i^k)^\top, (y_i^k)^\top]^\top$, $i \in [m]$, and up to a suitable termination:

$$x_i^{k+1} \approx_{\mathcal{M}} \arg\min_{x\in\mathbb{R}^d}\ u^k(x), \quad \forall i \in [m]; \tag{P^k}$$

`(S.2) Extrapolation step:`

$$z_i^{k+1} = x_i^{k+1} + \frac{\alpha^k(1-\alpha^k)}{(\alpha^k)^2 + \alpha^{k+1}}\left(x_i^{k+1} - x_i^k\right), \quad \forall i \in [m]. \tag{6}$$

end for

3.3 Convergence of DCatalyst: Strongly convex losses

Our analysis begins assessing the convergence of DCatalyst's outer loop (Proposition 1). Here, a suitable warm-restart mechanism and termination criterion for $\mathcal{M}$ are provided, which maintain the inexactness of solving the proximal subproblems in (S.1) within the threshold ensuring convergence of the outer loop. Our second result outlines specific tuning recommendations for the decentralized algorithm $\mathcal{M}$ used in the inner loop, designed to meet the established termination criterion (Proposition 2). Finally, combining both the outer- and inner-loop convergence, we obtain the overall computation/communication complexity of DCatalyst (Theorem 3). The proofs of the results in this section can be found in Sec. 6.

Notation and probability space: We denote the state variables generated by the decentralized algorithm $\mathcal{M}$ at the inner iteration t of the outer iteration k by $\boldsymbol{s}^{k,t}$. For the subproblem (P^k), we denote its solution as $x^{k,\star}$ and $\mathbf{x}^{k,\star} := 1_m(x^{k,\star})^\top$. When solving the subproblem (P^k) using the chosen algorithm $\mathcal{M}$, we refer to $\mathcal{L}^k : \mathbb{R}^{m\times(d+d')} \to \mathbb{R}_+$ as the merit function associated with $\mathcal{M}$ when applied to (P^k) and satisfying Assumption 4. Also, with a slight abuse of notation, we will write $r_{\mathcal{M},\delta}$ instead of $r_{\mathcal{M}}$ in (4), to explicitly highlight the dependency of the convergence rate on the design parameter δ.

When $\mathcal{M}$ is a randomized algorithm, the sequences generated by Algorithm 1 are random, and the following probability space is subsumed. Define $T_k \in (0,\infty)$ as the number of (inner) iterations performed by the decentralized algorithm $\mathcal{M}$ at the outer iteration k. Considering $\{\{\boldsymbol{s}^{k,t}\}_{t=0}^{T_k}\}_{k\geq 0} \in \prod_0^\infty \mathbb{R}^{m\times(d+d')}$ as the sequence of random variables generated by Algorithm 1, let Ω denote the sample space of all the sample paths $\omega := \{\{\underset{\sim}{\boldsymbol{s}}^{k,t}\}_{t=0}^{T_k}\}_{k\geq 0} \in \prod_0^\infty \mathbb{R}^{m\times(d+d')}$, where $\underset{\sim}{\boldsymbol{s}}^{k,t}$ is a realization of $\boldsymbol{s}^{k,t}$. For each k, define

$\Omega_k := \prod_0^{T_k} \mathbb{R}^{m\times(d+d')}$, and let $\mathcal{B}_k$ be the Borel σ-algebra of $\prod_{\ell=0}^{k} \Omega_\ell$. Suppose $\mathcal{M}$ induces a sequence of Borel probability measures $\{P_k\}_{k=0}^{\infty}$, where each P_k is defined on $\mathcal{B}_k$, satisfying the property: for each $E \in \mathcal{B}_k$ and any $n \geq 0$, $P_k(E) = P_{k+n}\left(E \times \prod_{\ell=k+1}^{k+n} \Omega_\ell\right)$. By the Ionescu–Tulcea's theorem (Klenke, 2008), there exists a unique probability measure $\mathbb{P}$ on $\mathcal{F} := \sigma\left(\cup_{k=0}^{\infty} \cup_{E_k\in\mathcal{B}_k} \left(E_k \times \prod_{\ell=k+1}^{\infty} \Omega_\ell\right)\right)$, which agrees with all P_k's: for any $E \in \mathcal{B}_k$, $\mathbb{P}\left(E \times \prod_{\ell=k+1}^{\infty} \Omega_\ell\right) = P_k(E)$. The probability space of interest is then $(\Omega, \mathbb{P}, \mathcal{F})$. Further, we define the filtration $\{\mathcal{F}_k\}_{k\geq 0}$, where $\mathcal{F}_{k+1} := \sigma\left(\{\{\boldsymbol{s}^{\ell,t}\}_{t=0}^{T_\ell}\}_{\ell=0}^{k}\right) = \mathcal{B}_k \times \prod_{\ell=k+1}^{\infty} \Omega_\ell$, and $\mathcal{F}_0 := \{\emptyset, \Omega\}$. All inequalities involving conditional expectations with respect to $\mathcal{F}_k$ are understood to hold almost surely, though this specification is omitted for brevity.

Our first result is the convergence of outer loop of DCatalyst, as given next.

Proposition 1 (Convergence of the outer loop) *Consider Problem* (P) *under Assumption 1 with $\mu > 0$. Let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1 in the following setting:* **(i)** *in Step (S.1), each subproblem* (P^k) *is (approximately) solved by a decentralized algorithm $\mathcal{M}$, which satisfies Assumption 4, with merit function denoted by $\mathcal{L}^k : \mathbb{R}^{m\times(d+d')} \to \mathbb{R}_+$;* **(ii)** *in Step (S.2), α^k is chosen as $\alpha^k = \alpha := \sqrt{\mu/(\mu+\delta)}$, for all $k \geq 0$;* **(iii)** *for each outer loop $k \geq 0$, $\mathcal{M}$ is terminated after T_k (inner) iterations, such that*

$$\mathbb{E}[\mathcal{L}^k(\boldsymbol{s}^{k,T_k})] \leq \epsilon_0(1-c\alpha)^{k+1}, \tag{7}$$

where $c \in (0,1)$ is some universal constant, and $\epsilon_0 > 0$ is arbitrarily chosen; and **(iv)** *the warm-restart $\boldsymbol{x}^k := \boldsymbol{x}^{k-1,T_{k-1}}$ is used.*

Then, for all $k \geq 0$,

$$\frac{1}{m}\mathbb{E}[\|\boldsymbol{x}^k - \boldsymbol{x}^\star\|^2] \leq c_{scvx}\,(1-c\alpha)^k\,, \text{ with } c_{scvx} = \mathcal{O}\left(\left(\frac{\delta^2}{\mu^2}+1\right)(\|\boldsymbol{x}^\star - \boldsymbol{x}^0\|^2 + \epsilon_0)\right). \tag{8}$$

The explicit expression of c_{scvx} can be found in the proof [see eq. (38)*, Sec. 6].*

Notice that, under Assumption 4, T_k is well-defined for any $k \geq 0$, that is, $T_k < \infty$.

Proposition 1 establishes linear convergence of the outer loop: For any given $\epsilon > 0$, $(1/m)\mathbb{E}[\|\mathbf{x}^k - \mathbf{x}^\star\|^2] \leq \epsilon$, in $k \geq K$ (outer) iterations, where

$$K = c\alpha \log\frac{c_{scvx}}{\epsilon} = \mathcal{O}\left(\sqrt{\frac{\mu+\delta}{\mu}}\log\frac{(1+\delta^2/\mu^2)(\|\mathbf{x}^\star - \mathbf{x}^0\|^2+\epsilon_0)}{\epsilon}\right).$$

The total number of inner-plus-outer iterations, is then

$$N_{\mathtt{it}} = \sum_{k=0}^{K} T_k.$$

We provide next sufficient conditions for T_k to be *uniformly* bounded with respect to k.

Assumption 5 *Given $\mu > 0$ (resp. $\mu = 0$), instate the setting of Proposition 1 (resp. Proposition 4). The following holds for each $k \geq 0$:*

$$\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0}) \leq c_{\mathcal{M}}\,\mathcal{L}^k(\boldsymbol{s}^{k,T_k}) + \frac{d_{\mathcal{M}}}{m}\|\boldsymbol{z}^k - \boldsymbol{z}^{k+1}\|^2,$$

where $c_{\mathcal{M}}, d_{\mathcal{M}} > 0$ are some problem-dependent constants independent of k.

At a high level, the assumption necessitates that the merit function at the start of each new inner loop does not significantly deviate from its value at the end of the preceding inner loop. In Sec. 4, we demonstrate through several examples that this condition can be satisfied by properly designing a warm-start initialization of the state variables (in particular the y-variables) in the inner-loop algorithm $\mathcal{M}$.

Equipped with Assumption 5, we prove the following uniform upper bound for T_k.

Proposition 2 *Instate the setting of Proposition 1, and additionally suppose that Assumption 5 holds. Then, the minimal T_k for condition (7) to hold is bounded by*

$$T_k \leq \left\lceil r_{\mathcal{M},\delta} \cdot \max\left\{ \log \frac{c_{\mathcal{L}} \mathcal{L}^0(\boldsymbol{s}^{0,0})}{\epsilon_0(1-c\alpha)},\ \log \frac{c_{\mathcal{L}} c_{\mathcal{M}} + 36 d_{\mathcal{M}} c_{scvx} \frac{1}{\epsilon_0(1-c\alpha)^2}}{1-c\alpha} \right\} \right\rceil = \widetilde{\mathcal{O}}(r_{\mathcal{M},\delta}).$$

Here, $r_{\mathcal{M},\delta}$ corresponds to $r_{\mathcal{M}}$ in (4) with $\mathcal{M}$ now applied to minimize u^k in (5) (hence depending also on δ).

Decentralized algorithms employed in the inner loop may require multiple rounds of communications/iteration. Denote such a number by $N_{\mathtt{com},\mathcal{M}}$, the total number $N_{\mathtt{com}}$ of communication rounds is

$$N_{\mathtt{com}} = N_{\mathtt{it}} \cdot N_{\mathtt{com},\mathcal{M}}.$$

Combining Proposition 1 and Proposition 2, we obtain the desired result for the total number of iterations and communication cost of Algorithm 1.

Theorem 3 *Given Problem (P) under Assumption 1 with $\mu > 0$, let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1 in the following setting:* **(i)** *in Step (S.1), a decentralized algorithm $\mathcal{M}$ is employed, satisfying Assumption 4 with merit function denoted by $\mathcal{L}^k : \mathbb{R}^{m\times(d+d')} \to \mathbb{R}_+$;* **(ii)** *in Step (S.2), α^k is chosen as $\alpha^k = \alpha := \sqrt{\mu/(\mu+\delta)}$, for all $k \geq 0$;* **(iii)** *for each outer loop $k \geq 0$, $\mathcal{M}$ is terminated after T_k inner iterations, such that (7) holds with $\mathcal{L}^k$ as defined in (i); and* **(iv)** *the warm-restart $\boldsymbol{x}^k := \boldsymbol{x}^{k-1,T_{k-1}}$ is used and Assumption 5 holds. Then,*

$$\frac{1}{m}\mathbb{E}[\|\boldsymbol{x}^k - \boldsymbol{x}^\star\|^2] \leq \epsilon,$$

after N_{it} inner-plus-outer iterations given by

$$N_{it} = \widetilde{\mathcal{O}}\left(r_{\mathcal{M},\delta} \sqrt{\frac{\mu+\delta}{\mu}} \log \frac{(1+\delta^2/\mu^2)(\|\boldsymbol{x}^\star - \boldsymbol{x}^0\|^2 + \epsilon_0)}{\epsilon} \right). \tag{9}$$

The total number of communications is $N_{com} = N_{it} \cdot N_{com,\mathcal{M}}$.

Theorem 3 certifies linear convergence of DCatalyst to the solution of (P). The iteration/communication complexity, as outlined in (9), is influenced by the tunable parameter δ and the convergence rate $r_{\mathcal{M},\delta}$ of the chosen algorithm $\mathcal{M}$. A crucial aspect of this theorem is providing tuning recommendations for δ, potentially achieving(near-)optimal communication complexity (up to log-factors). Sec. 4 provides specific choices for δ to enhance convergence of a range of existing, non-accelerated, decentralized algorithms.

3.4 Convergence of DCatalyst: Convex losses

This section studies convergence of Algorithm 1 when applied to (non-strongly) convex functions u. We mirror the structure of Sec. 3.3.

Our first result establishes convergence of the outer loop.

Proposition 4 (Convergence of the outer loop) *Consider Problem (P) under Assumption 1 with $\mu = 0$. Let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1 in the following setting:* **(i)** *in Step (S.1), a decentralized algorithm $\mathcal{M}$ satisfying Assumption 4 with merit function $\mathcal{L}^k : \mathbb{R}^{m\times(d+d')} \to \mathbb{R}_+$ is employed, to (approximately) solve the subproblem (P^k);* **(ii)** *in Step (S.2), α^k is chosen recursively according to*

$$(\alpha^k)^2 = (1-\alpha^k)(\alpha^{k-1})^2,\ \forall k \geq 1, \quad \textit{and} \quad \alpha^0 = \frac{\sqrt{5}-1}{2}; \tag{10}$$

(iii) *for each outer loop $k \geq 0$, $\mathcal{M}$ is terminated after T_k iterations, such that*

$$\mathbb{E}[\mathcal{L}^k(\boldsymbol{s}^{k,T_k})] \leq \left(\frac{1}{k+3}\right)^{4+2r_0} \epsilon_0, \tag{11}$$

where $r_0 > 0$ is some universal constant, and $\epsilon_0 > 0$ is arbitrarily chosen; and **(iv)** *the warm-restart $\boldsymbol{x}^k := \boldsymbol{x}^{k-1,T_{k-1}}$ is used, for all $k \geq 1$.*

Then, for all $k \geq 0$,

$$\frac{1}{2\delta m}\sum_{i=1}^{m}\mathbb{E}[\|\nabla M_{\frac{1}{\delta}u}(x_i^k)\|^2] \leq \frac{c_{cvx}}{(k+2)^2}, \quad \textit{with} \quad c_{cvx} = \mathcal{O}(\delta(\|\boldsymbol{x}^\star - \boldsymbol{x}^0\|^2 + \epsilon_0)).$$

The explicit expression of c_{cvx} is given in (39), Sec. 6.

Moreover, if $r \equiv 0$, for all $k \geq 0$, we have

$$\frac{1}{2\delta m}\sum_{i=1}^{m}\mathbb{E}[\|\nabla u(x_i^k)\|^2] \leq \frac{c_{cvx}(L+\delta)^2}{\delta^2(k+2)^2}.$$

The following result provides the growth rate of T_k.

Proposition 5 *Instate the setting of Proposition 4; suppose that* **(i)** *Assumption 5 holds, and* **(ii)** *the iterates $\{\mathbf{x}^k\}$ are bounded, $\|x_i^k\| \leq B$, for all $i \in [m]$ and $k \geq 0$, and some $B \in (0,\infty)$. Then, the minimal T_k for (11) to hold is bounded:*

$$\begin{aligned} T_k &\leq \left\lceil r_{\mathcal{M},\delta} \cdot \max\left\{\log\frac{c_{\mathcal{L}}9^{r_0+2}\mathcal{L}^0(\boldsymbol{s}^{0,0})}{\epsilon_0}, \log\left(c_{\mathcal{L}}c_{\mathcal{M}}2^{4+2r_0} + \frac{36c_{\mathcal{L}}d_{\mathcal{M}}B^2(k+3)^{4+2r_0}}{\epsilon_0}\right)\right\}\right\rceil \\ &= \widetilde{\mathcal{O}}\left(r_{\mathcal{M},\delta}\log k\right). \end{aligned} \tag{12}$$

Combining Proposition 4 and Proposition 5, we obtain the total number of inner-plus-outer iterations and communications to ϵ-optimality.

Theorem 6 *Given Problem (P) under Assumption 1 with $\mu = 0$, let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1 in the following setting:* **(i)** *in Step (S.1), a decentralized algorithm $\mathcal{M}$ satisfying Assumption 4 is employed;* **(ii)** *in Step (S.2), α^k is chosen recursively according to (10);* **(iii)** *for each outer loop $k \geq 0$, $\mathcal{M}$ is terminated after T_k inner iterations, such that (11) holds; and* **(iv)** *the warm-restart $\boldsymbol{x}^k := \boldsymbol{x}^{k-1,T_{k-1}}$ is used, Assumption 5 holds, and in addition $\{\boldsymbol{x}^k\}$ are bounded. Then, for all $k \geq 0$,*

$$\frac{1}{2\delta m}\sum_{i=1}^{m}\mathbb{E}[\|\nabla M_{\frac{1}{\delta}r}(x_i^k)\|^2] \leq \epsilon,$$

or when $r \equiv 0$,

$$\frac{\delta}{2m(L+\delta)^2}\sum_{i=1}^{m}\mathbb{E}[\|\nabla u(x_i^k)\|^2] \leq \epsilon,$$

after N_{it} inner-plus-outer iterations given by

$$N_{it} = \widetilde{\mathcal{O}}\left(r_{\mathcal{M},\delta}\sqrt{\frac{\delta(\|\boldsymbol{x}^\star - \boldsymbol{x}^0\|^2 + \epsilon_0)}{\epsilon}}\log\frac{\delta(\|\boldsymbol{x}^\star - \boldsymbol{x}^0\|^2 + \epsilon_0)}{\epsilon}\right).$$

The total number of communications is $N_{com} = N_{it} \cdot N_{com,\mathcal{M}}$.

We note that our findings diverge from those of the Catalyst framework in the centralized setting, e.g., (Lin et al., 2015, 2018). Beyond extending to the decentralized setting, a critical distinction lies in the nature of the termination time T_k, which is deterministic in our approach, unlike the stochastic nature of those in the literature, which predict termination in expectation. This makes our termination more practical.

The boundedness of the sequence is a standard assumption for (nonstrongly) convex losses, which is implied e.g. by the compactness of $\mathtt{dom}\, r$–a frequent constraint in machine learning applications that helps manage solution complexity. Alternatively, Theorem 6 may be reformulated under the coerciveness of u rather than boundedness of the iterates–another conditions widely used in the machine learning community (e.g., Lin et al. (2015, 2018); Li and Lin (2020)). In this case, T_k would be defined such that $\mathcal{L}^k(\boldsymbol{s}^{k,T_k}) \leq \epsilon_{k+1}$, making it a random variable. We can then demonstrate that $\mathbb{E}[T_k]$ is of the same order of the RHS of (12). We omit further details because of page limits.

4. Applications of the DCatalyst Framework

We apply DCatalyst to three representative decentralized algorithms: SONATA (Sun et al., 2022), PUDA (Alghunaim et al., 2020), and PMGT-LSVRG (Ye et al., 2026). This selection is motivated by the distinctive features of each of these algorithms. **(i)** SONATA is applicable to both strongly convex and convex (possibly composed) functions u, and can exploit function similarity to enhance convergence rates. By integrating it with DCatalyst, we aim to inherit these features while achieving accelerated rates. **(ii)** PUDA is representative of a variety of (primal-dual) (proximal) decentralized algorithms, which includes others such as EXTRA (Shi et al., 2015), DIGing (Nedić et al., 2017), and Exact Diffusion (Yuan et al., 2019)–employing DCatalyst provides a unified approach to accelerating all these

methods. **(iii)** For scenarios where each f_i has a finite-sum structure (satisfying Assumption 2), we apply DCatalyst to accelerate PMGT-LSVRG (Ye et al., 2026), marking the first accelerated variance-reduction decentralized algorithm applicable to composite functions u. Another by-product of our framework is DCatalyst-VR-EXTRA, which is obtained by applying DCatalyst to VR-EXTRA (Li et al., 2022).

To evaluate the performance of DCatalyst, in addition to the total (i.e., inner-plus-outer) communications towards ϵ-optimality, we report also the total computational complexity, measured by the total number of (proximal) gradient updates conducted by each agent.

4.1 DCatalyst SONATA (Sun et al., 2022)

The SONATA algorithm (Sun et al., 2022) applied to Problem (P) reads: for each $i \in [m]$ and $t \geq 0$ (and proper initialization),

$$\begin{aligned} x_i^{t+1/2} &= \arg\min_{x\in\mathbb{R}^d} \tilde{f}_i(x; x_i^t) + \left\langle y_i^t - \nabla f_i(x_i^t), x - x_i^t \right\rangle + r(x), \\ x_i^{t+1} &= \sum_{j=1}^m w_{ij} x_j^{t+1/2}, \\ y_i^{t+1} &= \sum_{j=1}^m w_{ij} \left(y_j^t + \nabla f_j(x_j^{t+1}) - \nabla f_j(x_j^t) \right). \end{aligned}$$

Here, $\tilde{f}_i$ is a local surrogate of f_i, chosen to exploit the potential structure of f_i; we will consider explicitly the following two instances of $\tilde{f}_i$:

$$\tilde{f}_i(x; x_i^t) = \begin{cases} f_i(x) + \dfrac{\beta}{2} \left\| x - x_i^t \right\|^2, & \text{(SONATA-F, under Ass. 3);} \\ f_i(x_i^t) + \left\langle \nabla f_i(x_i^t), x - x_i^t \right\rangle + \dfrac{L}{2} \left\| x - x_i^t \right\|, & \text{(SONATA-L).} \end{cases}$$

The first choice retains the entire function f_i, which will be shown being particularly suitable to exploit function similarity among the f_i (specifically, under Assumption 3), while the second one retains only the first-order information, generally leading to prox-friendly subproblems. The gossip weights, w_{ij}, are chosen such that (i) $w_{ij} > 0$ if $(i,j) \in \mathcal{G}$, and $w_{ij} = 0$ otherwise; and (ii) $W = [w_{ij}]_{i,j=1}^m \in \mathbb{R}^{m\times m}$ is doubly stochastic, i.e., $1_m^\top W = 1_m^\top$ and $W 1_m = 1_m$. We denote $\rho := \left\| W - (1/m) 1_m 1_m^\top \right\| < 1$. Notice that several rules have been proposed for the choice of such a W; example include the Laplacian, the Metropolis-Hasting, and the maximum-degree weight rules (Xiao et al., 2005; Nedić et al., 2018).

We apply the DCatalyst framework to SONATA-F and SONATA-L, which serve as the algorithm $\mathcal{M}$ in the inner loop of Algorithm 1. In the following, we outline the specific tuning of these algorithms to ensure they satisfy all the assumptions required for the application of Theorem 3 and Theorem 6. Throughout this section, when using SONATA-F we will implicitly assume similarity among the f_i's, as stipulated by Assumption 3.

• **On Assumption 4:** The result below summarizes the convergence property of SONATA-F and SONATA-L, in agreement with Assumption 4.

Lemma 7 *Consider SONATA applied to Problem (P) under Assumption 1 (with $\mu > 0$) with the following initialization: $x_i^0 \in \mathtt{dom} r$ and $y_i^0 = \nabla f_i(x_i^0)$, for all $i \in [m]$. Suppose*

$$\rho^2 \leq \begin{cases} \dfrac{\mu^2\beta^2}{5712(L+\beta)^2\left(9\beta^2+4\left(L+\beta\right)^2\right)} = \mathcal{O}\left(\left(\dfrac{\beta/\mu}{\kappa}+1\right)^4\right), & (\textit{SONATA-F}); \\ \min\left\{\dfrac{\mu^2}{13440L_{\max}^2}, \dfrac{L^2}{12L^2+84L_{\max}^2}\right\} = \mathcal{O}\left(\kappa_\ell^{-2}\right), & (\textit{SONATA-L}). \end{cases}$$

Then, Assumption 4 holds with the following positions:

- *Lyapunov function:*

$$\mathcal{L}(\boldsymbol{s}) = \frac{2}{\mu m}\sum_{i=1}^m \left(u(x_i)-u^\star\right) + \frac{\eta}{m}\left(4L_{\max}^2\|\boldsymbol{x}_\perp\|^2 + 2\|\boldsymbol{y}_\perp\|^2\right),$$

  where $\boldsymbol{s} = [\boldsymbol{x}, \boldsymbol{y}]$,

$$\boldsymbol{x}_\perp := \boldsymbol{x} - 1_m(\bar{x})^\top,\ \boldsymbol{y}_\perp := \boldsymbol{y} - 1_m(\bar{y})^\top,\ \textit{and } \eta = \begin{cases} \dfrac{34}{\mu(16\beta+\mu)}, & (\textit{SONATA-F}), \\ \dfrac{10}{\mu(4L+\mu)}, & (\textit{SONATA-L}). \end{cases}$$

- *Linear convergence: the contraction property (4) holds with*

$$c_{\mathcal{L}} = 1 \quad \textit{and} \quad r_{\mathcal{M}} = \begin{cases} 2+\dfrac{32\beta}{\mu} = \mathcal{O}\left(\dfrac{\beta}{\mu}\right), & (\textit{SONATA-F}); \\ 2+\dfrac{8L}{\mu} = \mathcal{O}\left(\kappa_g\right), & (\textit{SONATA-L}). \end{cases}$$

• **On Assumption 5:** When embedded in the inner loop of Algorithm 1 to minimize u^k, and initialized at the beginning of the $(k+1)$th inner loop as

$$\begin{aligned} &\mathbf{x}^{k+1} = \mathbf{x}^{k,T_k}, \\ &\mathbf{y}^{k+1} = \mathbf{y}^{k,T_k} + \delta\left(\mathbf{z}^{k+1} - \mathbf{z}^k\right), \end{aligned} \quad \text{with} \quad \delta = \begin{cases} \beta-\mu, & \text{(SONATA-F)}; \\ L-\mu, & \text{(SONATA-L)}, \end{cases} \tag{13}$$

SONATA satisfies Assumption 5, as proved next.

Lemma 8 *Consider SONATA applied to the subproblem (P^k), with initialization (13) and tuning as in Lemma 7 (applied here to u^k). Assumption 5 holds with the following positions:*

$$c_{\mathcal{M}} = 2 \textit{ and } d_{\mathcal{M}} = \frac{2\delta^2}{(\delta+\mu)^2} + 16\eta\delta^2, \textit{ with } \eta := \begin{cases} \dfrac{34}{(\mu+\delta)(16\beta+\mu+\delta)}, & (\textit{SONATA-F}), \\ \dfrac{10}{(\mu+\delta)(4L+\mu+5\delta)}, & (\textit{SONATA-L}). \end{cases}$$

Proof See Appendix B.1. ■

• **Convergence of DCatalyst-SONATA:** Since both Assumptions 4 and 5 are satisfied, convergence of DCatalyst, equipped with SONATA in the inner loop, comes readily from Theorems 3 (for strongly convex u) and 6 (for convex u), as summarized below.

Corollary 9 *Given Problem (P) under Assumption 1 with $\mu > 0$, let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1 in the following setting:* **(i)** *in Step (S.1), either SONATA-L or SONATA-F is employed, with initialization as in (13) and tuning as in Lemma 7 (applied to u^k);* **(ii)** *in Step (S.2), α^k is chosen as $\alpha^k = \alpha := \sqrt{\mu/(\mu+\delta)}$, for all $k \geq 0$;* **(iii)** *for each outer loop $k \geq 0$, SONATA is terminated after T_k inner iterations, given by*

$$T_k = \left\lceil r_{\mathcal{M},\delta} \log \frac{2 + \frac{36}{(1-c\,\alpha)^2}\left(\frac{2\delta^2}{(\delta+\mu)^2} + 16\eta\delta^2\right)\left(2 + \frac{\delta}{\mu} + \frac{(2m+1)(\mu+\delta)^2}{\mu^2(1-c)} + \frac{2\sqrt{2000}(\mu+\delta)^2\sqrt{m}}{\mu^2(1-c)^2}\right)}{1-c\alpha} \right\rceil,$$

where η is the same as in Lemma 8 and

$$r_{\mathcal{M},\delta} = \begin{cases} 2 + \dfrac{32\beta}{\mu+\delta} = 34, & (\textit{DCatalyst-SONATA-F}); \\ 2 + \dfrac{8(L+\delta)}{\mu+\delta} \leq 18, & (\textit{DCatalyst-SONATA-L}). \end{cases} \tag{14}$$

Then, $(1/m)\,\|\boldsymbol{x}^k - \boldsymbol{x}^\star\|^2 \leq \epsilon$ after a total number of communication rounds and computations given respectively by

$$\begin{cases} \widetilde{\mathcal{O}}\left(\sqrt{\dfrac{\beta}{\mu(1-\rho)}}\log\dfrac{1}{\epsilon}\right) \quad \textit{and} \quad \widetilde{\mathcal{O}}\left(\sqrt{\dfrac{L+\beta}{\beta}}\cdot\dfrac{\beta}{\mu}\log^2\dfrac{1}{\epsilon}\right), & (\textit{DCatalyst-SONATA-F}); \\ \widetilde{\mathcal{O}}\left(\sqrt{\dfrac{\kappa_g}{1-\rho}}\log\dfrac{1}{\epsilon}\right) \quad \textit{and} \quad \widetilde{\mathcal{O}}\left(\sqrt{\kappa_g}\log\dfrac{1}{\epsilon}\right), & (\textit{DCatalyst-SONATA-L}). \end{cases}$$

Proof See Appendix B.2. ■

Corollary 10 *Given Problem (P) under Assumption 1 with $\mu = 0$, let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1 in the following setting:* **(i)** *in Step (S.1), either SONATA-L or SONATA-F is employed, with initialization as in (13) and tuning as in Lemma 7 (applied to u^k);* **(ii)** *in Step (S.2), α^k is chosen recursively according to (10);* **(iii)** *the iterates are bounded, $\|x_i^k\| \leq B$ for any $i \in [m]$, $k \geq 0$ and some $B \in (0,\infty)$;* **(iv)** *for each outer loop $k \geq 0$, SONATA is terminated after T_k inner iterations, given by*

$$T_k = \left\lceil r_{\mathcal{M},\delta} \log\left(2^{5+2r_0} + 1224(k+3)^{4+2r_0}\right)\right\rceil = \widetilde{\mathcal{O}}(\log k).$$

Then, $(1/(2\delta m)) \cdot \sum_{i=1}^m \|\nabla M_{\frac{1}{\delta}u}(x_i^K)\|^2 \leq \epsilon$, after a total number of communication rounds and computations respectively given by

$$\begin{cases} \widetilde{\mathcal{O}}\left(\sqrt{\dfrac{\beta}{(1-\rho)\epsilon}}\log\dfrac{1}{\epsilon}\right) & \text{and} \quad \widetilde{\mathcal{O}}\left(\sqrt{\dfrac{L+\beta}{\epsilon}}\log^2\dfrac{1}{\epsilon}\right), \qquad \textit{(DCatalyst-SONATA-F)}; \\ \widetilde{\mathcal{O}}\left(\sqrt{\dfrac{L}{(1-\rho)\epsilon}}\log\dfrac{1}{\epsilon}\right) & \text{and} \quad \widetilde{\mathcal{O}}\left(\sqrt{\dfrac{L}{\epsilon}}\log\dfrac{1}{\epsilon}\right), \qquad \textit{(DCatalyst-SONATA-L)}. \end{cases}$$

Proof See Appendix B.3. ■

Corollary 9 and 10 demonstrate that DCatalyst-SONATA-L can achieve optimal performance in terms of both communication and computational complexities, for composite functions u. By exploiting function similarity, DCatalyst-SONATA-F achieves (nearly) optimal communication complexity (Table 1) $\widetilde{\mathcal{O}}((1/\sqrt{1-\rho})\cdot\sqrt{\beta/\mu}\cdot\log(1/\epsilon))$, which compare favorably with existing algorithms, especially when f is ill-conditioned.

4.2 Accelerating PUDA (Alghunaim et al., 2020)

The PUDA algorithm applied to Problem (P) (with $\mu_{\min} > 0$) reads: for any $t \geq 0$ (with proper initialization of $\mathbf{x}^0, \mathbf{y}^0 \in \mathbb{R}^{m\times d}$),

$$\begin{aligned} \mathbf{y}^{t+1/2} &= (I-C)\mathbf{x}^t - \eta\nabla F(\mathbf{x}^t) - H\mathbf{y}^t, \\ \mathbf{y}^{t+1} &= \mathbf{y}^t + H\mathbf{y}^{t+1/2}, \\ \mathbf{x}^{t+1} &= \mathbf{prox}_{\eta r}(W\mathbf{y}^{t+1/2}), \end{aligned} \tag{15}$$

where $\nabla F(\mathbf{x}^k) \in \mathbb{R}^{m\times d}$ denotes $[\nabla f_1(x_1^k), \cdots, \nabla f_m(x_m^k)]^\top$, and the choice of $W, H, C \in \mathbb{R}^{m\times m}$ determines the specific algorithm for consideration. For instance, EXTRA (Shi et al., 2015) and Prox-ED (Alghunaim et al., 2020) are special instances of PUDA–see (Alghunaim et al., 2020) for details. To ensure that DCatalyst is compatible with the diverse algorithms encompassed by PUDA, we do not restrict the structure of these matrices but rather adhere to the conditions for PUDA's convergence as outlined in (Alghunaim et al., 2020), namely: (i) W is symmetric and doubly stochastic; (ii) H and C are symmetric, gossip matrices satisfying $H\mathbf{x}$ (resp.$C\mathbf{x}$) $= 0 \Leftrightarrow \left(I - \frac{1}{m}1_m 1_m^\top\right)\mathbf{x} = 0$; and (iii) W, H, and C satisfy $W^2 \preceq I - H^2, \quad 0 \preceq C \preceq 2I$.

We show next that Theorem 3 and 6 are applicable to state convergence of DCatalyst when PUDA is used an decentralized algorithm to solve the inner subproblems (P^k).

• **On Assumption 4:** In the setting above, and under a suitable tuning, PUDA applied to Problem (P) (under $\mu_{\min} > 0$) satisfies Assumption 4.

Lemma 11 *Consider PUDA applied to Problem (P) under Assumption 1 (with $\mu_{\min} > 0$), with stepsize $\eta = (2-\sigma_{\max}(C))/(2L_{\max})$, and initialization: $x_i^0 \in \mathbf{dom} r$, and $y_i^0 \in \mathbf{range}(H)$, for all $i \in [m]$. Then, Assumption 4 holds with the following positions:*

- *Lyapunov function:*

$$\mathcal{L}(\boldsymbol{s}) := \frac{1}{m}\sum_{i=1}^m \|x_i - x^\star\|^2 + \frac{1}{m}\sum_{i=1}^m \|y_i - y^\star\|^2,$$

where $\boldsymbol{s} = [\boldsymbol{x}, \boldsymbol{y}]$, and $y^\star$ is the unique limit point of $\{y_i^k\}_{k\geq 0}$ (belonging to `range`*(H)) (Alghunaim et al., 2020)).*

- *Linear convergence: the contraction property (4) holds with:*

$$c_{\mathcal{L}} = 1 \quad \textit{and} \quad r_{\mathcal{M}} = \max\left\{\frac{4L_{\max}}{\left(2-\sigma_{\max}(C)\right)^2\mu_{\min}}, \frac{1}{\sigma_{\min}^+(H^2)}\right\}.$$

• **On Assumption 5:** When embedding PUDA in the inner loop (S.1) of Algorithm 1, the selection of δ and the initialization of the $(k+1)$th inner loop are:

$$\begin{aligned}\mathbf{x}^{k+1} &= \mathbf{x}^{k,T_k},\\ \mathbf{y}^{k+1} &= \mathbf{y}^{k,T_k},\end{aligned} \quad \text{with} \quad \delta = \frac{4\sigma_{\min}^+(H^2)\left(L_{\max}-\mu_{\min}\right)}{\left(2-\sigma_{\max}(C)\right)^2 - 4\sigma_{\min}^+(H^2)} - \mu_{\min}, \tag{16}$$

then PUDA satisfies Assumption 5, as shown below.

Lemma 12 *With the initialization and selection of δ as in (16), Assumption 5 holds with*

$$c_{\mathcal{M}} = 2 \quad \textit{and} \quad d_{\mathcal{M}} = 2 + \frac{\sigma_{\max}(H^2)\left(9+9\delta^2\eta^2\right)}{\sigma_{\min}^+(H^2)},$$

where η denotes the stepsize of PUDA with the tuning as in Lemma 11 (but applied to u^k).

Proof See Appendix B.4. ■

• **Convergence of DCatalyst-PUDA:** Since both of Assumption 4 and 5 hold, we can apply Theorem 3 to assess the convergence property of DCatalyst equipped with PUDA.

Corollary 13 *Given Problem (P) under Assumption 1 with $\mu_{\min} > 0$, let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1 in the following setting:* **(i)** *in Step* `(S.1)`*, PUDA is employed, satisfying (16) and tuning as in Lemma 11 (applied to u^k);* **(ii)** *in Step* (`S.2`)*, α^k is chosen as $\alpha^k = \alpha := \sqrt{\mu_{\min}/(\mu_{\min}+\delta)}$, for all $k \geq 0$;* **(iii)** *for each outer loop $k \geq 0$, PUDA is terminated after T_k inner iterations, given by*

$$T_k = \left\lceil r_{\mathcal{M},\delta}\log\frac{2+\frac{36}{(1-c\,\alpha)^2}\left(2+\frac{\sigma_{\max}(H^2)\left(9+9\delta^2\eta^2\right)}{\sigma_{\min}^+(H^2)}\right)\left(2+\frac{\delta}{\mu_{\min}}+\frac{(2m+1)(\mu_{\min}+\delta)^2}{\mu_{\min}^2(1-c)}+\frac{2\sqrt{2000}(\mu_{\min}+\delta)^2\sqrt{m}}{\mu_{\min}^2(1-c)^2}\right)}{1-c\alpha}\right\rceil,$$

where η is the same as in Lemma 12, and $r_{\mathcal{M},\delta} = 1/\sigma_{\min}^+(H^2)$. Then, $(1/m)\left\|\boldsymbol{x}^k - \boldsymbol{x}^\star\right\|^2 \leq \epsilon$ after a total number of communication rounds and computations given by

$$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_\ell}{\left(\left(2-\sigma_{\max}(C)\right)^2 - 4\sigma_{\min}^+(H^2)\right)\sigma_{\min}^+(H^2)}}\log\frac{1}{\epsilon}\right).$$

Proof See Appendix B.5. ■

To further comment the convergence results above, let us consider a specific admissible choice of the weight matrices within the PUDA framework, as used in the Prox-ED algorithm (Alghunaim et al., 2020), namely: $W = (I + \tilde{W})/2$, $H^2 = (I - \tilde{W})/2$, and $C = 0$, where $\tilde{W} \in \mathbb{R}^{m\times m}$ is symmetric, doubly stochastic and primitive matrix matching the graph $\mathcal{G}$, with associated $\rho = \|\tilde{W} - (1/m)1_m 1_m^\top\| < 1$. The iteration and communication complexity of the plain Prox-ED reads

$$\mathcal{O}\left(\left(\kappa_\ell + \frac{1}{1-\rho}\right)\log\frac{1}{\epsilon}\right).$$

By applying our acceleration framework, DCatalyst-Prox-ED improves to

$$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_\ell}{1-\rho}}\log\frac{1}{\epsilon}\right).$$

This shows a more favorable dependence on the agents' losses condition numbers κ_ℓ and network connectivity ρ. Yet, the dependence on the condition number is more favorable in DCatalyst-SONATA-L (see Corollary 9), with the latter being the global condition number κ_g rather than the local one κ_ℓ.

4.3 Accelerating PMGT-LSVRG (Ye et al., 2026)

We apply the DCatalyst framework to the classes of (P) where the agent losses have an additive separable structure, as specified in (1) (Assumption 2). We bring acceleration to the variance reduction decentralized algorithm PMGT-LSVRG (Ye et al., 2026).

PMGT-LSVRG applied to Problem (P) reads: for any $t \geq 0$ (and proper initialization),

$$\begin{aligned}
&\mathbf{x}^{t+1} = \texttt{FastMix}\left(\mathbf{prox}_{\eta r}\left(\mathbf{x}^t - \eta\mathbf{y}^t\right), N_{\texttt{FM}}\right),\\
&\text{Each agent } i \in [m] \text{ computes:}\\
&\quad v_i^{t+1} = \begin{cases} x_i^t, & \text{with probability } \dfrac{1}{n},\\ v_i^t, & \text{otherwise;}\end{cases}\\
&\quad \tilde{g}_i^{t+1} = \begin{cases} \nabla f_i(x_i^t), & \text{if } v_i^{t+1} = x_i^t,\\ \tilde{g}_i^t, & \text{otherwise;}\end{cases}\\
&\quad \text{Pick } j_i \in [n] \text{ with probability } p_{ij_i} = \dfrac{L_{ij_i}}{\sum_j^n L_{ij}}, \text{ and update}\\
&\quad g_i^{t+1} = \frac{1}{np_{ij_i}}\left(\nabla f_{ij_i}(x_i^{t+1}) - \nabla f_{ij_i}(v_i^{t+1})\right) + \tilde{g}_i^{t+1},\\
&\mathbf{y}^{t+1} = \texttt{FastMix}\left(\mathbf{y}^t + \mathbf{g}^{t+1} - \mathbf{g}^t, N_{\texttt{FM}}\right).
\end{aligned} \tag{17}$$

Here, `FastMix` is defined in Algorithm 2 below, used for accelerating the consensus steps (Liu and Morse, 2011); $N_{\texttt{FM}}$ is a parameter representing the number of communications per iteration, and $\rho = \left\|W - (1/m)1_m 1_m^\top\right\|$, associated with the gossip matrix $W \in \mathbb{R}^{m\times m}$ defined as in Sec. 4.1. In PMGT-LSVRG, the x-and y-variables are the decision and tracking variables. The gradient tracking mechanism employs a variance reduction technique via the

auxiliary variables $\tilde{g}_i$'s and v_i's, whereby at each iteration the full batch gradient $\nabla f_i(x_i)$ is computed only with probability $1/n$ and v_i represents the most recent iterate at which $\nabla f_i(\cdot)$ is evaluated. The number of communications per iteration is counted as $N_{\texttt{com},\mathcal{M}} = 2N_{\texttt{FM}}$.

Algorithm 2 $\texttt{FastMix}\big(\mathbf{x}^0, N_{\texttt{FM}}\big)$

Input: $\mathbf{x}^0 = (x_i^0)_{i\in[m]}$, $N_{\texttt{FM}}$; ρ; **Output**: $\mathbf{x}^{N_{\texttt{pm}-1}}$; **Set**: $\mathbf{x}^{-1} := \mathbf{x}^0$ and $\eta := \frac{1-\sqrt{1-\rho^2}}{1+\sqrt{1-\rho^2}}$;
for $t = 0, 1, 2, \cdots, N_{\texttt{FM}} - 1$ **do**
$\mathbf{x}^{t+1} = (1+\eta)W\mathbf{x}^t - \eta\mathbf{x}^{t-1}$;
end for

Remark 14 *Differently from the original PMGT-LSVRG (Ye et al., 2026), we introduced here a variant where the index j_i in (17) is selected with probability $p_{ij_i} = (L_{ij_i})/\sum_j^n L_{ij}$ rather than uniform. As it will be showed in the convergence results below, this improves the rate dependence of the algorithm on the condition number, from $\tilde{\kappa}_s$ to κ_s.*

• **On Assumption 4:** The result below summarizes the convergence of PMGT-LSVRG, in agreement with Assumption 4. The proof builds on (Ye et al., 2026) and is omitted.

Lemma 15 *Consider the PMGT-LSVRG applied to Problem (P) under Assumption 1 ($\mu > 0$) and Assumption 2, with the following initialization: for all $i \in [m]$, $x_i^0 = v_i^0 \in \texttt{dom}r$; g_i^0 and $\tilde{g}_i^0$ are unbiased estimators of $\nabla f_i^0(x_i^0)$ and $\nabla f_i^0(v_i^0)$, respectively; and $(1/m)\sum_{i=1}^m y_i^0$ is an unbiased estimator of $(1/m)\sum_{i=1}^m \nabla f_i(x_i^0)$. Further, $\eta = 1/(16L_{\max})$ and $N_{\texttt{FM}} = (1/\sqrt{1-\rho}) \cdot \log\left(36\max\{6\kappa_s, n\}\right)$. Then, Assumption 4 holds with following positions:*

- *Lyapunov function:*

$$\mathcal{L}(\boldsymbol{s}) := \frac{1}{c_{\texttt{pm}}}\left(\|\bar{x} - x^\star\|^2 + 4n\eta^2\Delta_f\right) + \frac{1}{m}\left(\|\boldsymbol{x}_\perp\|^2 + \eta^2\|\boldsymbol{y}_\perp\|^2\right),$$

  where $\boldsymbol{s} := [\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{g}, \boldsymbol{v}, \tilde{\boldsymbol{g}}]$, $c_{\texttt{pm}} = 20r_{\texttt{pm}}/(1 - 40r_{\texttt{pm}}\rho_{\texttt{pm}}^2)$, $r_{\texttt{pm}} = \max\{12\kappa_s, 2n\}$, $\rho_{\texttt{pm}} = (1-\sqrt{1-\rho})^{N_{\texttt{FM}}}$,

$$\Delta_f := \frac{1}{mn}\sum_{i=1}^m\sum_{j=1}^n \frac{1}{np_{ij}}\|\nabla f_{ij}(v_i) - \nabla f_{ij}(x^\star)\|^2,\ \boldsymbol{x}_\perp := \boldsymbol{x} - 1_m(\bar{x})^\top,\ \boldsymbol{y}_\perp := \boldsymbol{y} - 1_m(\bar{y})^\top.$$

- *Linear convergence: the contraction (4) holds with $c_{\mathcal{L}} = 1$ and $r_{\mathcal{M}} = 4r_{\texttt{pm}}$.*

• **On Assumption 5:** Suppose PMGT-LSVRG is embedded in the inner loop of Algorithm 1, with the following choice of δ and warm-start for the $(k+1)$th inner loop:

$$\begin{aligned}&\mathbf{x}^{k+1} = \mathbf{x}^{k,T_k},\ \mathbf{v}^{k+1} = \mathbf{v}^{k,T_k}, \mathbf{y}^{k+1} = \mathbf{y}^{k,T_k} + \delta(\mathbf{z}^k - \mathbf{z}^{k+1}),\ \mathbf{g}^{k+1} = \mathbf{g}^{k,T_k} + \delta(\mathbf{z}^k - \mathbf{z}^{k+1}),\\ &\tilde{\mathbf{g}}^{k+1} = \tilde{\mathbf{g}}^{k,T_k} + \delta(\mathbf{z}^k - \mathbf{z}^{k+1}), \quad \text{and} \quad \delta = \frac{\bar{L}_{\max}}{n} - \mu.\end{aligned} \tag{18}$$

Then, PMGT-LSVRG satisfies Assumption 5, as proved below.

Lemma 16 *Consider PMGT-LSVRG applied to subproblem (P^k), with the initialization (18) and tunning as in Lemma 15 (applied to u^k). Assumption 5 holds with*

$$c_{\mathcal{M}} = 2, \quad \textit{and} \quad d_{\mathcal{M}} = 2 + 8\eta^2\delta^2 + \frac{8\eta^2(\bar{L}_{\max} + \delta)^2}{n^2}.$$

Proof See Appendix B.6. ■

• **Convergence of DCatalyst-PMGT-LSVRG:** Since both Assumptions 4 and 5 hold, we can apply Theorem 3 to assess the convergence of DCatalyst-PMGT-LSVRG.

Corollary 17 *Given Problem (P) under Assumption 1 with $\mu > 0$, let $\{\mathbf{x}^k\}$ be the sequence generated by Algorithm 1 in the following setting:* **(i)** *in Step (S.1), PMGT-LSVRG is employed, satisfying (18) and tuning as in Lemma 15 (applied to u^k);* **(ii)** *in Step (S.2), α^k is chosen as $\alpha^k = \alpha := \sqrt{\mu/(\mu+\delta)}$, for all $k \geq 0$;* **(iii)** *for each outer loop $k \geq 0$, PMGT-LSVRG is terminated after T_k inner iterations, given by*

$$T_k = \left\lceil r_{\mathcal{M},\delta} \log \frac{2 + \frac{90}{(1-c\,\alpha)^2}\left(2 + \frac{\delta}{\mu} + \frac{(2m+1)(\mu+\delta)^2}{\mu^2(1-c)} + \frac{2\sqrt{2000}(\mu+\delta)^2\sqrt{m}}{\mu^2(1-c)^2}\right)}{1-c\alpha} \right\rceil,$$

with $r_{\mathcal{M},\delta} = 48n$. Then, $(1/m)\,\mathbb{E}[\|\boldsymbol{x}^k - \boldsymbol{x}^\star\|^2] \leq \epsilon$ after a total number of communication rounds and a number of computations (in expectation) given respectively by

$$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_s n}{1-\rho}} \log \frac{1}{\epsilon}\right), \quad \textit{and} \quad \widetilde{\mathcal{O}}\left(\sqrt{\kappa_s n} \log \frac{1}{\epsilon}\right).$$

Proof See Appendix B.7. ■

Compared with PMGT-LSVRG, (reported in Table 3), DCatalyst-PMGT-LSVRG exhibits better performance: PMGT-LSVRG has communication complexity $\mathcal{O}\left((1/(\sqrt{1-\rho}))\right.$ $\left.(\tilde{\kappa}_s \log \tilde{\kappa}_s + n \log n) \cdot \log(1/\epsilon)\right)$ and computational complexity $\mathcal{O}\left((n + \tilde{\kappa}_s)\ \log(1/\epsilon)\right)$, while DCatalyst-PMGT-LSVRG's reads $\widetilde{\mathcal{O}}\left((1/\sqrt{1-\rho})\sqrt{\kappa_s n}\ \log(1/\epsilon)\right)$ and $\widetilde{\mathcal{O}}\left(\sqrt{\kappa_s n} \cdot \log(1/\epsilon)\right)$, showing significant savings in communication/computation especially for ill-conditioned problems. This sets a new benchmark for composite optimization functions. On the other hand, existing decentralized VR algorithms, such as Acc-VR-EXTRA-CA (Li et al., 2022), ADFS (Hendrikx et al., 2021) and DVR-Catalyst (Hendrikx et al., 2020a), have more favorable dependence on the condition number of f, namely: κ_ℓ-dependence versus κ_s in DCatalyst-PMGT-LSVRG. However, all these methods are applicable only to *smooth* functions u. DCatalyst-PMGT-LSVRG is the first decentralized algorithm merging acceleration and VR technique that is applicable to composite functions u (with f_i additively separable).

5. Inexact Estimating Sequences

This section introduces the tool of inexact estimating sequence–the analytic machinery underlying the DCatalyst framework. The key idea is to modify the very definition of estimating sequence (Nesterov, 2013; Baes, 2009; Lin et al., 2015, 2018; d'Aspremont et al.,

2021) so as to explicitly incorporate *exogenous* error sequences, leading to our notion of inexact estimating sequences. These errors later encode the behavior of different decentralized algorithms, allowing us to handle multiple agent-specific variable sequences and rigorously capture both consensus errors and inexact decentralized subproblem solutions in a unified way. Equipped with this new definition, we prove convergence of any algorithm falling within this framework directly "up to" a controllable exogenous error term, and we establish accelerated rates under suitable algorithmic tuning that keeps these errors in check. Because the errors are exogenous, the framework is algorithm-independent and can, in principle, be applied beyond DCatalyst, offering a tool of independent interest. In the next section, we use this inexact-estimating-sequence machinery to prove in particular, convergence of Algorithm 1. This stands in contrast with the original Catalyst analysis, which keeps Nesterov's classical definition of estimating sequences and then incorporates an *endogenous* error term tied to a specific inner-loop termination rule; as a consequence, its convergence guarantees are tightly coupled to that particular stopping criterion and cannot be used as a black-box tool for generic decentralized schemes.

Given Problem (P) (under Assumption 1) and the the Moreau envelope $\mathtt{M}_{\frac{1}{\delta}u} : \mathbb{R}^d \to \mathbb{R}$, we recall that $\mathtt{M}_{\frac{1}{\delta}u}$ is δ-smooth and $\mu_{\mathtt{M}}$-strongly convex, with $\mu_{\mathtt{M}} = (\delta\mu)/(\delta+\mu)$. Furthermore, notice that $\mathtt{M}^\star_{\frac{1}{\delta}u} := \mathtt{M}_{\frac{1}{\delta}u}(x^\star) = u^\star$, where $x^\star$ is a minimizer of u.

We are ready to state the definition of general inexact estimating sequences.

Definition 18 (Inexact estimating sequence) *An inexact estimating sequence is a tuple of a sequence of functions $\{(\psi_i^k)_{i\in[m]}\}_{k=0}^\infty$, a sequence of positive numbers $\{\alpha^k\}_{k=0}^\infty$, and a real-value error-sequence $\{(\epsilon_i^k)_{i\in[m]}\}_{k=0}^\infty$ such that*

(i) *$\alpha^k \in (0,1)$, for all $k \geq 0$, and $\lim_{k\to\infty}\prod_{t=0}^k(1-\alpha^t) = 0$;*

(ii) *each $\psi_i^k : \mathbb{R}^d \to \mathbb{R}$ satisfies*

$$\psi_i^{k+1}(x^\star) \leq (1-\alpha^k)\psi_i^k(x^\star) + \alpha^k\left(\mathtt{M}^\star_{\frac{1}{\delta}u} + \epsilon_i^k\right), \quad \forall k \geq 0.$$

The following result shows how inexact estimating sequences can be leveraged to guide the development of appropriate decentralized algorithms solving effectively Problem (P).

Lemma 19 *Let $\{(\psi_i^k)_{i\in[m]}\}_{k=0}^\infty$, $\{\alpha^k\}_{k=0}^\infty$, $\{(\epsilon_i^k)_{i\in[m]}\}_{k=0}^\infty$ be an inexact estimating sequence. Suppose there exists a sequence of iterates $\{(\tilde{x}_i^k)_{i\in[m]}\}_{k=0}^\infty$ such that, for all $k \geq 0$ and $i \in [m]$,*

$$\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k) \leq \left[\psi_i^{k,\star} := \min_{x\in\mathbb{R}^d}\psi_i^k(x)\right] + \epsilon_{\psi,i}^k, \tag{19}$$

for some $\{(\epsilon_{\psi,i}^k)_{i\in[m]}\}_{k=0}^\infty$, with $\epsilon_{\psi,i}^0 \equiv 0$. Then,

$$0 \leq \psi_i^k(x^\star) + \epsilon_{\psi,i}^k - \mathtt{M}^\star_{\frac{1}{\delta}u} \leq \lambda^k\left(\psi_i^0(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} + \sum_{t=0}^{k-1}\frac{\epsilon_{tot,i}^t}{\lambda^{t+1}}\right), \tag{20}$$

where

$$\lambda^k := \prod_{t=0}^{k-1}(1-\alpha^t) \quad \text{and} \quad \epsilon_{tot,i}^k := \epsilon_{\psi,i}^{k+1} - (1-\alpha^k)\epsilon_{\psi,i}^k + \alpha^k\epsilon_i^k. \tag{21}$$

Proof See Appendix A.1. ■

When the algorithm generating $\{(\tilde{x}_i^k)_{i\in[m]}\}_{k=0}^{\infty}$ is stochastic, $\{(\tilde{x}_i^k)_{i\in[m]}\}_{k=0}^{\infty}$ is a random sequence, and Lemma 19 is understood to hold for any realization of the random variables.

The subsequent proposition elucidates the convergence rate of decentralized algorithms producing iterates $\{(\tilde{x}_i^k)_{i\in[m]}\}_{k=0}^{\infty}$ conforming to Lemma 19. This convergence depends on the decay rate of the error sequences $\mathbb{E}[|\epsilon_{tot,i}^k|]$.

Proposition 20 *Let $\{(\tilde{x}_i^k)_{i\in[m]}\}_{k=0}^{\infty}$ be a sequence satisfying Lemma 19.*

(i) *$\mu > 0$: Suppose $(1/m)\sum_{i=1}^m \mathbb{E}\big[|\epsilon_{tot,i}^k|\big] \le C_{scvx}\,(1-c\alpha)^k$, for all $k \ge 0$ and for some problem-dependent constant C_{scvx}. Then,*

$$\frac{1}{m}\mathbb{E}[\|\tilde{\boldsymbol{x}}^k - \boldsymbol{x}^\star\|^2] \le c_{scvx}(1-c\alpha)^k,$$

where

$$c_{scvx} = \frac{1}{m}\sum_{i=1}^m \left(\frac{2}{\mu_M}\left(\psi_i^0(x^\star) - M^\star_{\frac{1}{\delta}u} + \frac{C_{scvx}}{\alpha(1-c)}\right)\right).$$

(ii) *$\mu = 0$: Suppose $(1/m)\sum_{i=1}^m \mathbb{E}\big[|\epsilon_{tot,i}^k|\big] \le C_{cvx}/(k+1)^{3+r_0}$, for all $k \ge 0$ and for some problem-dependent constant C_{cvx}. Then,*

$$\frac{1}{2\delta m}\sum_{i=1}^m \mathbb{E}[\|\nabla M_{\frac{1}{\delta}u}(\tilde{x}_i^k)\|^2] \le \frac{c_{cvx}}{(k+2)^2}, \tag{22}$$

where

$$c_{cvx} = \frac{1}{m}\sum_{i=1}^m \left(\psi_i^0(x^\star) - M^\star_{\frac{1}{\delta}u} + \frac{10C_{cvx}}{r_0}\right).$$

Moreover, if $r \equiv 0$, then

$$\frac{1}{2\delta m}\sum_{i=1}^m \mathbb{E}[\|\nabla u(\tilde{x}_i^k)\|^2] \le \frac{c_{cvx}(L+\delta)^2}{\delta^2(k+2)^2}. \tag{23}$$

Proof See Appendix A.2. ■

5.1 A constructive approach to a family of inexact estimating sequences

We provide a constructive approach to derive an explicit class of inexact estimating sequences. which leads to a wide range of decentralized designs. Our method unfolds in two steps: **(i)** we first specify a family of inexact estimating sequence, parametrized by certain free quantities (see Lemma 21 below); **(ii)** we then exploit the degrees of freedom offered by this family to satisfy the remaining condition (19).

Lemma 21 *Assume, for any $i \in [m]$, and $k \ge 0$,*

(i) $\psi_i^0(\bullet) : \mathbb{R}^d \to \mathbb{R}$ *is an arbitrary convex function on* $\mathbb{R}^d$*;*

(ii) $\{(\tilde{z}_i^k)_{i\in[m]}\}_{k=0}^{\infty}$ *and* $\{(e_i^k)_{i\in[m]}\}_{k=0}^{\infty}$ *are arbitrary sequences in* $\mathbb{R}^d$*;*

(iii) $\{\alpha^k\}_{k=0}^{\infty}$ *is chosen according to Definition 18(ii).*

(iv) $\{(\psi_i^k)_{i\in[m]}\}_{k=0}^{\infty}$ *is defined recursively as*

$$\psi_i^{k+1}(x) := (1-\alpha^k)\psi_i^k(x) + \alpha^k\left(M_{\frac{1}{\delta}u}(\tilde{z}_i^k) + \langle \nabla M_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k, x - \tilde{z}_i^k\rangle + \frac{\mu_M}{2}\|x-\tilde{z}_i^k\|^2\right). \tag{24}$$

Then, the tuple $\{(\psi_i^k)_{i\in[m]}\}_{k=0}^{\infty}$, $\{\alpha^k\}_{k=0}^{\infty}$, $\{(\epsilon_i^k)_{i\in[m]}\}_{k=0}^{\infty}$, *with* $\epsilon_i^k := \langle e_i^k, x^\star - \tilde{z}_i^k\rangle$ *forms an inexact estimating sequence.*

Given the above family of inexact estimating sequences, we have two control sequences to choose, $\{(\tilde{z}_i^k)_{i\in[m]}\}_k$ and $\{(\tilde{x}_i^k)_{i\in[m]}\}_k$, to maintain recursively the relation (19), for a suitable sequence $\{\epsilon_{\psi,i}^k\}_k$. According to Lemma 21, we are also free in the choice of ψ_i^0. Following (Nesterov, 2013), we choose it as a simple quadratic function:

$$\psi_i^0(x) = \psi_i^{0,*} + \frac{\zeta^0}{2}\|x - v_i^0\|^2, \tag{25}$$

for some $\psi_i^{0,*} \in \mathbb{R}$, $\zeta^0 \in (0,\infty) \in \mathbb{R}^d$, and $v_i^0 \in \mathbb{R}^d$. To obtain $\psi_i^{k,\star}$ as well as further specify the recursion of $\{\epsilon_{\psi,i}^k\}_{k\geq 0}$ in (19), we proceed writing $\{(\psi_i^k(x))_{i\in[m]}\}_k$ in the canonical form.

Lemma 22 (Canonical form) *Let* $\psi_i^0(x)$ *be given as in (25). The process (24) preserves the canonical form of functions* $\left\{(\psi_i^k)_{i\in[m]}\right\}_k$*:*

$$\psi_i^k(x) = \psi_i^{k,\star} + \frac{\zeta^k}{2}\|x - v_i^k\|^2, \quad \forall i \in [m],\ k \geq 0,$$

where the sequences $\{\zeta^k\}$, $\{v_i^k\}$, *and* $\{\psi_i^{k,\star}\}$ *are defined as follows:*

$$\begin{aligned}
\zeta^{k+1} &= (1-\alpha^k)\zeta^k + \alpha^k\mu_M,\\
v_i^{k+1} &= \frac{(1-\alpha^k)\zeta^k}{\zeta^{k+1}}v_i^k + \frac{\alpha^k\mu_M}{\zeta^{k+1}}\tilde{z}_i^k - \frac{\alpha^k}{\zeta^{k+1}}\left(\nabla M_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k\right),\\
\psi_i^{k+1,\star} &= (1-\alpha^k)\psi_i^{k,\star} + \alpha^k M_{\frac{1}{\delta}u}(\tilde{z}_i^k) - \frac{(\alpha^k)^2}{2\zeta^{k+1}}\|\nabla M_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k\|^2\\
&\quad + \frac{\alpha^k(1-\alpha^k)\zeta^k}{\zeta^{k+1}}\left(\left\langle \nabla M_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k, v_i^k - \tilde{z}_i^k\right\rangle + \frac{\mu_M}{2}\|\tilde{z}_i^k - v_i^k\|^2\right).
\end{aligned} \tag{26}$$

The proof of this result is standard (Nesterov, 2013), hence omitted.

We are left to enforce (19), which holds under the following propositions.

Lemma 23 (On condition (19)) *In the setting above, let for each* $i \in [m]$*,*

(i) $\psi_i^{0,\star} = M_{\frac{1}{\delta}u}(\tilde{x}_i^0)$ *and* $v_i^0 = \tilde{x}_i^0$*;*

(ii) $\tilde{z}_i^0 = \tilde{x}_i^0$, *and*

$$\tilde{z}_i^{k+1} := \tilde{x}_i^{t+1} + \frac{\alpha^k(1-\alpha^k)}{(\alpha^k)^2 + \alpha^{k+1}}(\tilde{x}_i^{k+1} - \tilde{x}_i^k); \tag{27}$$

(iii) *and*

$$e_i^k := \delta(\tilde{z}_i^k - \tilde{x}_i^{k+1}) - \nabla M_{\frac{1}{\delta}u}(\tilde{z}_i^k). \tag{28}$$

Then, (19) holds, with

$$\epsilon_{\psi,i}^{k+1} := (1-\alpha^k)\epsilon_{\psi,i}^k + (1-\alpha^k)\left\langle e_i^k, \tilde{x}_i^k - \tilde{z}_i^k\right\rangle + \left\langle e_i^k + \nabla M_{\frac{1}{\delta}u}(\tilde{z}_i^k), \frac{1}{\delta}e_i^k\right\rangle, \text{ and } \epsilon_{\psi,i}^0 = 0. \tag{29}$$

Proof The lemma is proved by induction. Notice that (19) holds for $k=0$, and $\epsilon_{\psi,i}^0 = 0$. Suppose (19) hold for a given $k > 0$. Then, in view of (26) and $\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k) \geq \langle \nabla \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k), \tilde{x}_i^k - \tilde{z}_i^k\rangle + \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k)$, we have

$$\begin{aligned}
\psi_i^{k+1,\star} &\geq -(1-\alpha^k)\epsilon_{\psi,i}^k + \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k) + (1-\alpha^k)\left\langle \nabla \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k), \tilde{x}_i^k - \tilde{z}_i^k\right\rangle \\
&\quad - \frac{(\alpha^k)^2}{2\zeta^{k+1}}\|\nabla \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k\|^2 + (1-\alpha^k)\left\langle \nabla \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k, \frac{\alpha^k\zeta^k}{\zeta^{k+1}}\left(v_i^k - \tilde{z}_i^k\right)\right\rangle \\
&\overset{(a)}{=} -(1-\alpha^k)\epsilon_{\psi,i}^k + \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k) + (1-\alpha^k)\left\langle -e_i^k, \tilde{x}_i^k - \tilde{z}_i^k\right\rangle \\
&\quad - \frac{1}{2\delta}\|\nabla \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k\|^2 + (1-\alpha^k)\left\langle \nabla \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k, \frac{\alpha^k\zeta^k}{\zeta^{k+1}}\left(v_i^k - \tilde{z}_i^k\right) + \tilde{x}_i^k - \tilde{z}_i^k\right\rangle,
\end{aligned}$$

where in (a) we used $\zeta^{k+1} = \delta(\alpha^k)^2$ for all $k \geq 0$, which is a consequence of the recursion on ζ^{k+1} in (26), given $\zeta^0 = (\delta(\alpha^0)^2 - \alpha^0\mu_{\mathtt{M}})/(1-\alpha^0)$.

Chaining the above inequality with the following (due to (28) and δ-smoothness of $\mathtt{M}_{\frac{1}{\delta}u}$),

$$\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^{k+1}) \leq \mathtt{M}_{\frac{1}{\delta}u}(z_i^k) - \frac{1}{2\delta}\|\nabla \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k\|^2 + \frac{1}{\delta}\left\langle e_i^k, \nabla \mathtt{M}_{\frac{1}{\delta}u}(\tilde{z}_i^k) + e_i^k\right\rangle,$$

and using (27), yields (19), with $\epsilon_{\psi,i}^{k+1}$ as in (29). ■

6. Convergence Analysis of DCatalyst

We are now ready to prove convergence of Algorithm 1, leveraging the framework of inexact estimating sequences. Specifically, Sec. 6.1 establishes convergence of the outer-loop, showing that $\{(z_i^k)_{i\in[m]}\}_{k=0}^\infty$ and $\{(x_i^k)_{i\in[m]}\}_{k=0}^\infty$ are an instance of the sequences $\{(\tilde{z}_i^k)_{i\in[m]}\}_{k=0}^\infty$ and $\{(\tilde{x}_i^k)_{i\in[m]}\}_{k=0}^\infty$ The proof of Propositions 1 and 4 will then follow by checking the conditions on the decay of the error sequence $(1/m)\sum_{i=1}^m \mathbb{E}\left[|\epsilon_{tot,i}^k|\right]$ as in (i) and (ii) of Proposition 20, respectively. Subsequently, Sec. 6.2 is dedicated to the convergence analysis of the inner loop, proving Propositions 2 and 5. Finally, the proofs of Theorems 3 and 6 follow readily combining the results from both the outer- and inner-loop analyses.

6.1 Analysis of the outer-loop of Algorithm 1: Proof of Propositions 1 and 4

Setting $\tilde{z}_i^k := z_i^k$ and $\tilde{x}_i^k := x_i^k$ for all $i \in [m]$ and $k \geq 0$, where $\{(z_i^k)_{i\in[m]}\}_{k=0}^{\infty}$ and $\{(x_i^k)_{i\in[m]}\}_{k=0}^{\infty}$ are the sequences generated by Algorithm 1, we see that Algorithm 1 fits the setting of Lemma 23. Hence, Proposition 20 applies directly, provided we show that $\frac{1}{m}\sum_{i=1}^m \mathbb{E}\big[|\epsilon_{tot,i}^k|\big]$ satisfies the bound required in the statement of the proposition.

6.1.1 Bounding $\epsilon_{tot,i}^k$

Using (21) and (29),

$$\begin{aligned}\epsilon_{tot,i}^k &= \epsilon_{\psi,i}^{k+1} - (1-\alpha^k)\epsilon_{\psi,i}^k + \alpha^k \epsilon_i^k \\ &= \alpha^k \left\langle e_i^k, x^\star - z_i^k \right\rangle + (1-\alpha^k)\left\langle e_i^k, x_i^k - z_i^k \right\rangle + \frac{1}{\delta}\left\langle e_i^k, \nabla \mathtt{M}_{\frac{1}{\delta}u}(z_i^k) + e_i^k \right\rangle \\ &= \left\langle e_i^k, \alpha^k x^\star + (1-\alpha^k)x_i^k - x_i^{k+1} \right\rangle = \alpha^k \left\langle e_i^k, x^\star - v_i^{k+1} \right\rangle.\end{aligned}$$

Therefore,

$$\begin{aligned}\frac{1}{m}\sum_{i=1}^m \mathbb{E}[|\epsilon_{tot,i}^k|] &\leq \alpha^k \frac{1}{m}\sum_{i=1}^m \mathbb{E}[\|e_i^k\|\|x^\star - v_i^{k+1}\|] \\ &\leq \alpha^k \left(\frac{1}{m}\sum_{i=1}^m \mathbb{E}[\|e_i^k\|^2]\right)^{1/2} \left(\frac{1}{m}\sum_{i=1}^m \mathbb{E}[\|x^\star - v_i^{k+1}\|^2]\right)^{1/2}.\end{aligned} \tag{30}$$

We proceed bounding the two terms above.

1) Bounding $\left(\frac{1}{m}\sum_{i=1}^m \mathbb{E}[\|e_i^k\|^2]\right)^{1/2}$: For all $i \in [m]$ and $k \geq 0$, using (28) and $x^{k,\star} = \bar{z}^k - (1/\delta)\nabla \mathtt{M}_{\frac{1}{\delta}u}(\bar{z}^k)$,

$$e_i^k = \nabla \mathtt{M}_{\frac{1}{\delta}u}(\bar{z}^k) - \nabla \mathtt{M}_{\frac{1}{\delta}u}(z_i^k) + \delta(\bar{z}^k - z_i^k) + \delta(\bar{x}^{k+1} - x_i^{k+1}) + \delta(x^{k,\star} - \bar{x}^{k+1}).$$

Then, for any $k \geq 0$, setting $x_i^{-1} = 0$ (as a dummy variable), we have

$$\begin{aligned}&\|e_i^k\|^2 = \|\nabla \mathtt{M}_{\frac{1}{\delta}u}(\bar{z}^k) - \nabla \mathtt{M}_{\frac{1}{\delta}u}(z_i^k) + \delta(\bar{z}^k - z_i^k) + \delta(\bar{x}^{k+1} - x_i^{k+1}) + \delta(x^{k,\star} - \bar{x}^{k+1})\|^2 \\ &\leq 12\delta^2\|z_i^k - \bar{z}^k\|^2 + 3\delta^2\|\bar{x}^{k+1} - x_i^{k+1}\|^2 + 3\delta^2\|x^{k,\star} - \bar{x}^{k+1}\|^2 \\ &\leq 72\delta^2\|x_i^k - \bar{x}^k\|^2 + 48\delta^2\|x_i^{k-1} - \bar{x}^{k-1}\|^2 + 3\delta^2\|\bar{x}^{k+1} - x_i^{k+1}\|^2 + 3\delta^2\|x^{k,\star} - \bar{x}^{k+1}\|^2.\end{aligned} \tag{31}$$

We proceed distinguishing the two cases of $\mu > 0$ and $\mu = 0$.

(i) $\mu > 0$: By Assumption 4 and (7),

$$\frac{1}{m}\mathbb{E}[\|\mathbf{x}^{k+1} - \mathbf{x}^{k,\star}\|^2] \leq \mathbb{E}[\mathcal{L}^k(\boldsymbol{s}^{k,T_k})] \leq \epsilon_0(1-c\alpha)^{k+1}, \quad \forall k \geq 0.$$

When $k \geq 0$, using (31), yields

$$\mathbb{E}[\|e_i^k\|^2] \leq 72\delta^2(1-c\alpha)^k m\epsilon_0 + 48\delta^2(1-c\alpha)^{k-1}m\epsilon_0 + 3\delta^2(1-c\alpha)^{k+1}m\epsilon_0.$$

Then

$$\mathbb{E}[\|e_i^k\|^2] \leq 125\delta^2(1-c\alpha)^{k-1}m\epsilon_0, \quad \text{for any } i \in [m],\ k \geq 0. \tag{32}$$

(ii) $\mu = 0$, Invoking Assumption 4 and (11), and following similar steps as in the case $\mu > 0$, we obtain: for any $i \in [m]$ and $k \geq 0$,

$$\mathbb{E}[\|e_i^k\|^2] \leq 125\delta^2 m\epsilon_0 \left(\frac{1}{k+1}\right)^{4+2r_0}. \tag{33}$$

2) Bounding $\alpha^k \left(\frac{1}{m}\sum_{i=1}^m \mathbb{E}[\|x^\star - v_i^{k+1}\|^2]\right)^{1/2}$: By the canonical form of ψ_i^k and Lemma 23,

$$\begin{aligned}&\frac{\zeta^{k+1}}{2}\|x^\star - v_i^{k+1}\|^2 + \mathtt{M}_{\frac{1}{\delta}u}(x_i^{k+1}) - \mathtt{M}^\star_{\frac{1}{\delta}u} \quad \leq \frac{\zeta^{k+1}}{2}\|x^\star - v_i^{k+1}\|^2 + \psi_i^{k+1,\star} + \epsilon_{\psi,i}^{k+1} - \mathtt{M}^\star_{\frac{1}{\delta}u}\\ &= \psi_i^{k+1}(x^\star) + \epsilon_{\psi,i}^{k+1} - \mathtt{M}^\star_{\frac{1}{\delta}u}.\end{aligned}$$

By Lemma 19 , we have

$$\begin{aligned}&\frac{1}{\lambda^{k+1}}\left(\frac{\zeta^{k+1}}{2}\|x^\star - v_i^{k+1}\|^2 + \mathtt{M}_{\frac{1}{\delta}u}(x_i^{k+1}) - \mathtt{M}^\star_{\frac{1}{\delta}u}\right) \leq \psi_i^0(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} + \sum_{t=0}^k \frac{\alpha^t\|e_i^t\|\|x^\star - v_i^{t+1}\|}{\lambda^{t+1}}\\ &\leq \frac{\zeta^0+\delta}{2}\|x^\star - x_i^0\|^2 + \sum_{t=0}^k \frac{\sqrt{\zeta^{t+1}}\|e_i^t\|\|x^\star - v_i^{t+1}\|}{\sqrt{\delta}\lambda^{t+1}},\end{aligned}$$

where in the inequality we used $\zeta^{t+1} = \delta(\alpha^t)^2$.

Then,

$$\begin{aligned}&\mathbb{E}\left[\frac{\zeta^{k+1}}{2\lambda^{k+1}}\|x^\star - v_i^{k+1}\|^2\right]\\ &\leq \frac{\zeta^0+\delta}{2}\|x^\star - x_i^0\|^2 + \sum_{t=0}^k \mathbb{E}\left[\left(\sqrt{\frac{2}{\delta\lambda^{t+1}}}\|e_i^t\|\right)\left(\sqrt{\frac{\zeta_{t+1}}{2\lambda^{t+1}}}\|x^\star - v_i^{t+1}\|\right)\right]\\ &\leq \frac{\zeta^0+\delta}{2}\left\|x^\star - x_i^0\right\|^2 + \sum_{t=0}^k \left(\mathbb{E}\left[\frac{2}{\delta\lambda^{t+1}}\|e_i^t\|^2\right]\right)^{1/2}\left(\mathbb{E}\left[\frac{\zeta_{t+1}}{2\lambda^{t+1}}\|x^\star - v_i^{t+1}\|^2\right]\right)^{1/2}.\end{aligned} \tag{34}$$

We apply now (Schmidt et al., 2011, Lemma 1) to (34), and obtain

$$\mathbb{E}\left[\frac{\zeta^{k+1}}{2\lambda^{k+1}}\|x^\star - v_i^{k+1}\|^2\right] \leq \left(\sqrt{\frac{\zeta^0+\delta}{2}\|x^\star - x_i^0\|^2} + \sum_{t=0}^k \left(\mathbb{E}\left[\frac{2}{\delta\lambda^{t+1}}\|e_i^t\|^2\right]\right)^{1/2}\right)^2. \tag{35}$$

Next, we bound $\mathbb{E}[(\zeta^k/2)\|v_i^k - x^\star\|^2]$ separately for the two cases $\mu > 0$ and $\mu = 0$, utilizing the bounds of $\mathbb{E}[\|e_i^k\|^2]$ derived in (32) and (33), respectively.

(i) $\mu > 0$: Using $\alpha^k \equiv \alpha$, $\zeta^k \equiv \mu_{\mathtt{M}}$, and (35), we have

$$\begin{aligned}
\frac{1}{m}\sum_{i=1}^m \mathbb{E}\left[\frac{\mu_{\mathtt{M}}}{2}\|v_i^k - x^\star\|^2\right] &\leq \left(\sqrt{\frac{\mu_{\mathtt{M}}+\delta}{2}\|x^\star - x_i^0\|^2} + \sum_{t=0}^{k-1}\mathbb{E}\left[\left(\frac{\alpha\|e_i^t\|}{\sqrt{\lambda^{t+1}}}\sqrt{\frac{2}{\mu_{\mathtt{M}}}}\right)^2\right]^{1/2}\right)^2 \\
&\overset{(32)}{\leq} (1-c\alpha)^k \frac{1}{m}\sum_{i=1}^m \left(\sqrt{\frac{\mu_{\mathtt{M}}+\delta}{2}\|x^\star - x_i^0\|^2} + \frac{\alpha\sqrt{125\delta^2 m\epsilon_0}}{1-c\alpha}\sqrt{\frac{2}{\mu_{\mathtt{M}}}}\sum_{t=0}^{\infty}\left(\sqrt{\frac{1-\alpha}{1-c\alpha}}\right)^t\right)^2 \\
&= (1-c\alpha)^k \frac{1}{m}\sum_{i=1}^m \left(\sqrt{\frac{\mu_{\mathtt{M}}+\delta}{2}\|x^\star - x_i^0\|^2} + \sqrt{\frac{2}{\mu_{\mathtt{M}}(1-c\alpha)}}\frac{\alpha\sqrt{125\delta^2 m\epsilon_0}}{\sqrt{1-c\alpha}-\sqrt{1-\alpha}}\right)^2 \\
&\leq (1-c\alpha)^k \frac{1}{m}\sum_{i=1}^m \left((\mu_{\mathtt{M}}+\delta)\|x^\star - x_i^0\|^2 + \frac{2000\delta^2 m\epsilon_0}{\mu_{\mathtt{M}}(1-c)^2}\right).
\end{aligned} \tag{36}$$

Denote $c_{v,scvx} := \frac{1}{m}\sum_{i=1}^m \left((\mu_{\mathtt{M}}+\delta)\|x^\star - x_i^0\|^2 + \frac{2000\delta^2 m\epsilon_0}{\mu_{\mathtt{M}}(1-c)^2}\right)$.

(ii) $\mu = 0$: Using (35) and (33), yields

$$\begin{aligned}
\frac{1}{m}\sum_{i=1}^m \mathbb{E}\left[\frac{\zeta^k}{2}\|v_i^k - x^\star\|^2\right] &\leq \lambda^k \frac{1}{m}\sum_{i=1}^m \left(\sqrt{\frac{\zeta^0+\delta}{2}\|x^\star - x_i^0\|^2} + \sum_{t=0}^{k-1}\left(\mathbb{E}\left[\frac{2}{\delta\lambda^{t+1}}\|e_i^t\|^2\right]\right)^{1/2}\right)^2 \\
&\leq \lambda^k \frac{1}{m}\sum_{i=1}^m \left(\sqrt{\frac{\zeta^0+\delta}{2}\|x^\star - x_i^0\|^2} + \sum_{t=0}^{k-1}\sqrt{\frac{250\delta m\epsilon_0}{\lambda^{t+1}}}\left(\frac{1}{t+1}\right)^{2+r_0}\right)^2 \\
&\leq \lambda^k \frac{1}{m}\sum_{i=1}^m \left(\sqrt{\frac{\zeta^0+\delta}{2}\|x^\star - x_i^0\|^2} + \sum_{t=0}^{k-1} 3\sqrt{125\delta m\epsilon_0}\left(\frac{1}{t+1}\right)^{1+r_0}\right)^2 \\
&\leq \frac{4}{(k+2)^2}\frac{1}{m}\sum_{i=1}^m \left(\sqrt{\frac{\zeta^0+\delta}{2}\|x^\star - x_i^0\|^2} + 3\sqrt{125\delta m\epsilon_0}\left(\int_0^\infty \left(\frac{1}{t+1}\right)^{1+r_0}\mathrm{d}t + 1\right)\right)^2 \\
&= \frac{4}{(k+2)^2}\frac{1}{m}\sum_{i=1}^m \left(\sqrt{\frac{\zeta^0+\delta}{2}\|x^\star - x_i^0\|^2} + 3\sqrt{125\delta m\epsilon_0}\left(\frac{1}{r_0}+1\right)\right)^2 \\
&\leq \frac{1}{(k+2)^2}\frac{1}{m}\sum_{i=1}^m \left(8\delta\|x^\star - x_i^0\|^2 + 9000\delta m\epsilon_0\left(\frac{1}{r_0}+1\right)^2\right),
\end{aligned} \tag{37}$$

where in the last inequality we used $\zeta^0 = \delta$. Let $c_{v,cvx} := (1/m)\sum_{i=1}^m (8\delta\|x^\star - x_i^0\|^2 + 9000\delta m\epsilon_0(1/r_0+1)^2)$.

Equipped with the bound on $(1/m)\sum_{i=1}^m \mathbb{E}[|\epsilon_{tot,i}^k|]$, we are ready to prove convergence of the outer loop of Algorithm 1 by applying Proposition 20.

6.1.2 Application of Proposition 20

We separate the two cases $\mu > 0$ and $\mu = 0$.

(i) $\mu > 0$: Using (30), (32) and (36), we have

$$\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}[|\epsilon_{tot,i}^k|] \le \alpha\sqrt{\frac{\delta^2 m\epsilon_0 c_{v,scvx}}{\mu_{\mathtt{M}}}}(1-c\alpha)^{k-1}.$$

Denote $C_{scvx} := \alpha\sqrt{\frac{\delta^2 m\epsilon_0 c_{v,scvx}}{\mu_{\mathtt{M}}}}$. By Corollary 20, $(1/m)\mathbb{E}[\|\mathbf{x}^{k+1}-\mathbf{x}^\star\|^2] \le c_{scvx}(1-c\alpha)^{k+1}$, where

$$\begin{aligned}
c_{scvx} &= \frac{1}{m}\sum_{i=1}^{m}\left(\frac{2}{\mu_{\mathtt{M}}}\left(\psi_i^0(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} + \frac{C_{scvx}}{\alpha(1-c)}\right)\right)\\
&\le \frac{1}{m}\sum_{i=1}^{m}\left(\|x^\star - x_i^0\|^2 + \frac{2}{\mu_{\mathtt{M}}}(\mathtt{M}_{\frac{1}{\delta}u}(x_i^0) - \mathtt{M}^\star_{\frac{1}{\delta}u}) + \frac{2}{\mu_{\mathtt{M}}(1-c)}\sqrt{\frac{\delta^2 m\epsilon_0 c_{v,scvx}}{\mu_{\mathtt{M}}}}\right)\\
&\le \frac{1}{m}\sum_{i=1}^{m}\left(2+\frac{\delta}{\mu}\right)\|x^\star - x_i^0\|^2 + \frac{2}{\mu_{\mathtt{M}}(1-c)}\left(\sqrt{\frac{\delta^2\epsilon_0(\mu_{\mathtt{M}}+\delta)}{\mu_{\mathtt{M}}}\sum_{i=1}^{m}\|x-x_i^0\|^2} + \sqrt{\frac{2000\delta^4 m\epsilon_0^2}{\mu_{\mathtt{M}}^2(1-c)^2}}\right)\\
&\le \frac{1}{m}\sum_{i=1}^{m}\left(2+\frac{\delta}{\mu}\right)\|x^\star - x_i^0\|^2 + \frac{\delta(\mu_{\mathtt{M}}+\delta)}{\mu_{\mathtt{M}}^2(1-c)}\sum_{i=1}^{m}\|x^\star - x_i^0\|^2 + \left(\frac{\delta}{\mu_{\mathtt{M}}(1-c)} + \frac{2\sqrt{2000m}\delta^2}{\mu_{\mathtt{M}}^2(1-c)^2}\right)\epsilon_0.
\end{aligned} \tag{38}$$

(ii) $\mu = 0$: Following similar steps as for $\mu > 0$ and using (30), (33), and (37), we have

$$\begin{aligned}
\mathbb{E}[|\epsilon_{tot,i}^k|] &\le \alpha^k\sqrt{\frac{2}{\zeta^{k+1}}}\left(\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}[\|e_i^k\|^2]\right)^{1/2}\left(\frac{\zeta^{k+1}}{2m}\sum_{i=1}^{m}\mathbb{E}[\|x^\star - v_i^{k+1}\|^2]\right)^{1/2}\\
&\le \sqrt{250\delta m\epsilon_0 c_{v,cvx}}\left(\frac{1}{k+1}\right)^{3+r_0}.
\end{aligned}$$

Denote $C_{cvx} := \sqrt{250\delta m\epsilon_0 c_{v,cvx}}$. Applying Proposition 20, we obtain

$$\frac{1}{2\delta m}\sum_{i=1}^{m}\mathbb{E}[\|\nabla \mathtt{M}_{\frac{1}{\delta}u}(x_i^k)\|^2] \le \frac{c_{cvx}}{(k+1)^2} \text{ and } \frac{1}{2\delta m}\sum_{i=1}^{m}\mathbb{E}[\|\nabla u(x_i^k)\|^2] \le \frac{c_{cvx}(L+\delta)^2}{\delta^2(k+2)^2} \text{ (if } r\equiv 0),$$

where

$$c_{cvx} = \frac{4}{m}\sum_{i=1}^{m}\left(\psi_i^0(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} + \frac{10C_{cvx}}{r_0}\right) \le \frac{4}{m}\sum_{i=1}^{m}\left(\delta\|x^\star - x_i^0\|^2 + \frac{10C_{cvx}}{r_0}\right). \tag{39}$$

6.2 Analysis of the inner-loop of Algorithm 1: Proof of Propositions 2 and 5

We prove Propositions 1 and 4 by showing that the prescribed upper bounds on T_k therein ensure that (7) and (11) hold under Assumption 5, in the corresponding settings.

The key step in the inner-loop analysis is to control the merit function at the beginning of a new inner loop, $\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0})$. If $\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0})$ is sufficiently small, then the linear convergence of the inner algorithm implies that the required number of inner iterations T_k cannot be large. By Assumption 5, $\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0})$ does not deviate significantly from its value at the end of the previous inner loop, $\mathcal{L}^k(\boldsymbol{s}^{k,T_k})$, and the latter can be kept bounded via an induction argument. We treat separately the cases $\mu > 0$ and $\mu = 0$.

6.2.1 Inner loop analysis ($\mu > 0$): Proof of Proposition 2

For simplicity, denote

$$c_T := \left\lceil r_{\mathcal{M},\delta} \cdot \max\left\{ \log \frac{c_{\mathcal{L}} \mathcal{L}^0(\boldsymbol{s}^{0,0})}{\epsilon_0(1-c\alpha)},\ \log \frac{c_{\mathcal{L}} c_{\mathcal{M}} + 36 d_{\mathcal{M}} c_{scvx} \frac{1}{\epsilon_0(1-c\alpha)^2}}{1-c\alpha} \right\} \right\rceil .$$

We proceed by induction, with following induction hypothesis: given $k \geq 0$ and any $\ell \in [k]$, there exists $T_\ell \in (0, c_T]$ such that (7) holds for the ℓth outer loop with such T_ℓ.

- $k = 0$: It is sufficient to pick $T_0 = \left\lceil r_{\mathcal{M},\delta} \log\left((c_{\mathcal{L}} \mathcal{L}^0(\boldsymbol{s}^{0,0}))/(\epsilon_0(1-c\alpha)) \right) \right\rceil \leq c_T$, for (7) to hold, since

$$\mathbb{E}[\mathcal{L}^0(\boldsymbol{s}^{0,T_0})|\mathcal{F}_0] \leq c_{\mathcal{L}} \left(1 - \frac{1}{r_{\mathcal{M},\delta}}\right)^{T_0} \mathcal{L}^0(\boldsymbol{s}^{0,0}) \leq \epsilon_0(1-c\alpha).$$

- $k > 0$: Suppose the induction hypothesis holds at given k. we leverage Assumption 5 to bound $\mathbb{E}[\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0})]$.

  We bound

$$\begin{aligned} \|\mathbf{z}^{k+1} - \mathbf{z}^k\| &\leq \left(1 + \frac{1-\alpha}{1+\alpha}\right) \|\mathbf{x}^{k+1} - \mathbf{x}^k\| + \frac{1-\alpha}{1+\alpha} \|\mathbf{x}^k - \mathbf{x}^{k-1}\| \\ &\leq 3 \max\{\|\mathbf{x}^{k+1} - \mathbf{x}^k\|, \|\mathbf{x}^k - \mathbf{x}^{k-1}\|\}. \end{aligned} \tag{40}$$

  Using $\|\mathbf{x}^{k+1} - \mathbf{x}^k\|$ by $\|\mathbf{x}^{k+1} - \mathbf{x}^k\| \leq \|\mathbf{x}^{k+1} - \mathbf{x}^\star\| + \|\mathbf{x}^\star - \mathbf{x}^k\|$, we proceed bounding $\|\mathbf{x}^\star - \mathbf{x}^k\|$. According to the induction hypothesis, (7) holds for all $\ell \in [k]$. Therefore, invoking Proposition 1 we have

$$\mathbb{E}[\|\mathbf{x}^\star - \mathbf{x}^\ell\|^2] \overset{(8)}{\leq} m c_{scvx} (1-c\alpha)^\ell, \quad \forall \ell \in [k+1].$$

  This yields

$$\mathbb{E}[\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0})] \leq c_{\mathcal{M}} \epsilon_0 (1-c\alpha)^{k+1} + 36 d_{\mathcal{M}} c_{scvx} (1-c\alpha)^{k-1}.$$

  Choose

$$T_{k+1} = \left\lceil r_{\mathcal{M},\delta} \log \frac{c_{\mathcal{L}} c_{\mathcal{M}} + 36 d_{\mathcal{M}} c_{scvx} \frac{1}{\epsilon_0(1-c\alpha)^2}}{1-c\alpha} \right\rceil \leq c_T.$$

  Clearly, (7) holds with

$$\mathbb{E}[\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,T_{k+1}})] \leq c_{\mathcal{L}} \left(1 - \frac{1}{r_{\mathcal{M},\delta}}\right)^{T_{k+1}} \mathbb{E}[\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0})] \leq \epsilon_0 (1-c\alpha)^{k+2}.$$

  We proved that the minimal T_k for (7) to hold is upper bounded by c_T, for all $k \geq 0$.

□

6.2.2 Inner Loop Analysis ($\mu = 0$): proof of Proposition 5

The procedure is similar to that in Sec. 6.2.2. Let

$$c_{T_k} := \left\lceil r_{\mathcal{M},\delta} \cdot \max\left\{ \log \frac{c_{\mathcal{L}} 9^{r_0+2} \mathcal{L}^0(\boldsymbol{s}^{0,0})}{\epsilon_0}, \log\left(c_{\mathcal{L}} c_{\mathcal{M}} 2^{4+2r_0} + \frac{36 c_{\mathcal{L}} d_{\mathcal{M}} B^2 (k+3)^{4+2r_0}}{\epsilon_0} \right) \right\} \right\rceil.$$

The induction hypothesis reads: given $k \geq 0$ and $\ell \in [k]$, there exists $T_\ell \in (0, c_{T_\ell}]$ such that (11) holds for the ℓth outer loop with such T_ℓ.

- $k = 0$: Choose $T_0 = \lceil r_{\mathcal{M},\delta} \log\left((c_{\mathcal{L}} 9^{r_0+2} \mathcal{L}^0(\boldsymbol{s}^{0,0}))/\epsilon_0 \right) \rceil \leq c_{T_0}$. It holds

$$\mathbb{E}[\mathcal{L}^0(\boldsymbol{s}^{0,T_0}) | \mathcal{F}_0] \leq c_{\mathcal{L}} \left(1 - \frac{1}{r_{\mathcal{M},\delta}}\right)^{T_0} \mathcal{L}^0(\boldsymbol{s}^{0,0}) \leq \frac{1}{3^{4+2r_0}} \epsilon_0.$$

- $k > 0$: Using (40) and $\left\| x_i^k \right\| \leq B$, we have

$$\mathbb{E}\left[\frac{1}{m} \|\mathbf{z}^{k+1} - \mathbf{z}^k\|^2 \right] \leq 36 B^2, \quad \forall k \geq 0. \tag{41}$$

  Then, by Assumption 5 and (41),

$$\begin{aligned} \mathbb{E}[\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0})] &\leq c_{\mathcal{M}} \mathbb{E}[\mathcal{L}^k(\boldsymbol{s}^{k,T_k})] + \frac{d_{\mathcal{M}}}{m} \mathbb{E}[\|\mathbf{z}^k - \mathbf{z}^{k+1}\|^2] \\ &\leq c_{\mathcal{M}} \left(\frac{1}{k+3}\right)^{4+2r_0} \epsilon_0 + 36 d_{\mathcal{M}} B^2. \end{aligned}$$

  Choosing

$$T_{k+1} = \left\lceil r_{\mathcal{M},\delta} \log\left(c_{\mathcal{L}} c_{\mathcal{M}} 2^{4+2r_0} + \frac{36 c_{\mathcal{L}} d_{\mathcal{M}} B^2 (k+4)^{4+2r_0}}{\epsilon_0} \right) \right\rceil \leq c_{T_{k+1}},$$

  makes (11) hold, with

$$\mathbb{E}[\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,T_{k+1}})] \leq c_{\mathcal{L}} \mathbb{E}[\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0})] \left(1 - \frac{1}{r_{\mathcal{M},\delta}}\right)^{T_{k+1}} \leq \left(\frac{1}{k+4}\right)^{4+2r_0} \epsilon_0.$$

□

7. Numerical Experiments

This section presents experiments conducted on real datasets across three instances of Problem (P): strongly convex u, convex (non-strongly convex) u, and losses f_i with a finite-sum structure. Given that DCatalyst introduces acceleration for minimizing *composite* objective functions (i.e., $r \neq 0$) across all these classes for the first time, our experiments predominantly focus on these types of functions. To offer a comprehensive comparison against existing accelerated decentralized methods, which are generally designed for smooth, unconstrained optimization problems, we have included additional experiments for smooth instances of Problem (P) in the arXiv version of this paper.

Unless otherwise specified, the setup for all experiments is as follows: we simulate an Erdos-Renyi graph with $m = 30$ nodes (agents) and an edge probability of $p = 0.5$. The gossip weight matrix used in all the algorithms is Metropolis-Hasting weight matrix. For strongly convex problems, the optimality gap at iteration k is defined as $\frac{1}{m}\sum_{i=1}^m \|x_i^k - x^\star\|^2$, while for (non-strongly) convex functions,the gap is measured using $\frac{1}{m}\sum_{i=1}^m u(x_i^k) - u^\star$.

7.1 Strongly Convex objectives

Our first experiment concerns the logistic regression model with the elastic net sparsity-inducing regularization. Beyond solely ℓ_1 penalty (LASSO), the elastic net regularization is proposed for a better selection of groups of correlated variables in statistical learning problems, which can help improve the prediction accuracy on many real datasets–see, e.g., (Zou and Hastie, 2005; Tay et al., 2023). The formulation of interest corresponds to Problem (P) with

$$f_i(x) = \frac{1}{n}\sum_{j=1}^n \log(1+\exp(-b_{ij}\cdot\langle x, a_{ij}\rangle)) + \frac{\gamma}{2}\|x\|_2^2 \quad \text{and} \quad r(x) = \lambda\|x\|_1, \tag{42}$$

where $a_{ij} \in \mathbb{R}^d$ and $b_{ij} \in \{-1, 1\}$. Here, the data set $\{(a_{ij}, b_{ij})\}_{j=1}^n$ is assumed to be private and owned only by agent i. We use MNIST dataset from LIBSVM (Chang and Lin, 2011) of size $N = 60000$ and feature dimension $d = 784$.

We contrast the proposed DCatalyst-SONATA-L and DCatalyst-SONATA-F with DPAG (Ye et al., 2020), which is the only available decentralized algorithm applicable to nonsmooth objective functions. All algorithms were implemented according to their theoretical guidelines. For DCatalyst-SONATA-L (resp. DCatalyst-SONATA-F), we set $\delta = L - \mu$ (resp. $\delta = \beta - \mu$), the momentum parameter $\alpha = \sqrt{\mu/(\mu+\delta)}$, and the number of inner-loop iterations $T_k = \lceil \log(L/\mu) \rceil$ (resp. $T_k = \lceil \log(\beta/\mu) \rceil$), for all k. In DCatalyst-SONATA-L, the local agents' prox-updates are computed in closed-form while in DCatalyst-SONATA-F, the accelerated proximal gradient method is employed, rub up to a tolerance of 10^{-8}. For the DPAG, we followed the tuning recommendation as in (Ye et al., 2020).

• **On the linear convergence:** The initial experiment aims to validate the linear convergence rate predicted by our theoretical findings for this class of problems. In these experiments, we set in (42), $\lambda = 0.01$ and $\gamma = 0.5$, resulting in $\beta/\mu = 3.1$ and $\kappa_g = 20$. Figure 1 summarizes the comparison, plotting the optimality gap of each algorithm versus the number of total communications (subplot (a)) and the number of inner-plus-outer iterations (subplot (a)) (for DCatalyst-SONATA-F, this includes also the number of iterations run by the accelerated proximal gradient method employed locally to compute the local agents' prox-updates). All the schemes achieve linear convergence. Consistent with our theoretical predictions (see Corollary 9), DCatalyst-SONATA-F excels particularly when the local functions f_i exhibit some similarity, quantified by $1 + \beta/\mu < \kappa_g$, outperforming both DCatalyst-SONATA-L and DPAG (Ye et al., 2020), which do not leverage function similarity. Notably, DCatalyst-SONAT-L significantly outperforms DPAG (Ye et al., 2020) in terms of communication rounds while maintaining comparability in the number of iterations. This is quite remarkable considering that DPAG is a *single-loop* scheme, specifically designed for this class of problems while DCatalyst-SONAT-L is the result of a general unified framework applicable to a much larger class of objective functions.

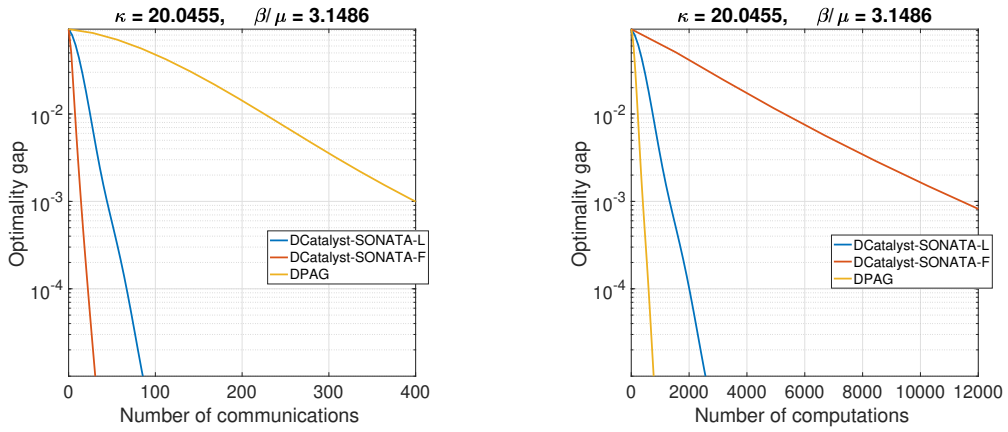

Figure 1: Comparison of distributed algorithms under strongly convex and non-smooth setting in **(a):** communication cost (left) and **(b):** computation cost (right).

• **On the impact of κ_g, β/μ, and total sample size.** We consider the following two scenarios. **(i)** *Changing β/μ with (almost) fixed κ_g*: We generate instances of logistic regression problem with decreasing β and (almost) fixed κ_g, increasing the local sample size n (starting from $n = 20$) while keeping $\gamma = 0.5$ fixed (and $\lambda = 0.01$), resulting in a $\kappa_g \approx 10.646$. Fig. 2 (left-panel) captures this scenario; we plot the number of communications to drive the optimality gap below 10^{-4}. In the mid-panel we report the same number of communications versus the total sample size N. **(ii)** *Changing κ_g with fixed β/μ*: We generate instances of logistic regression problem with varying κ_g, acting on γ, while keeping β/μ constant by changing the local sample size to compensate for the variation of μ via γ, resulting in $\beta/\mu \approx 15.53$. Fig. 2 (right-panel) captures this scenario; we plot the number of communications to drive the optimality gap below 10^{-4} versus κ_g, for fixed β/μ.

The following comments are in order. The left panel confirms what is predicted by Corollary 9: the convergence rate of DCatalist-SONATA-F scales with $\sqrt{\beta/\mu}$ while that of the other reported algorithms is almost invariant with β/μ. This is because those other methods use only gradient information and hence cannot benefit from statistical similarity. On the other hand, the right-panel shows that DCatalist-SONATA-L and DPAG exhibit a communication complexity that deteriorates when κ_g grows (of the order of $\sqrt{\kappa_g}$) whereas that of DCatalist-SONATA-F remains almost invariant. This is exactly what our theoretical results predicted. Notice also that both instances of the proposed framework uniformly outperform DPAG, in any simulated setting.

7.2 (Non-strongly) Convex Setting

As (non strongly) convex, nonsmooth instance of Problem (P), we consider the decentralized logistic regression problem with ℓ_1-regularization. This corresponds to the formulation in (42), with $\gamma = 0$. We set $\lambda = 10^{-4}$.

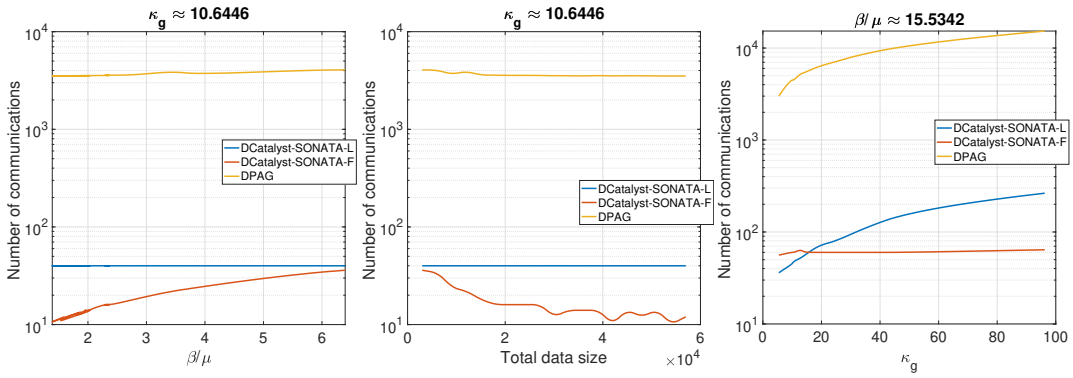

Figure 2: Comparison of distributed algorithms under strongly convex and non-smooth setting on the influence of parameters **(a):** similarity β/μ (left), **(b):** total sample size N (middle) and **(c):** global condition number κ_g to the number communication rounds needed to reach a precision of 10^{-4} (right).

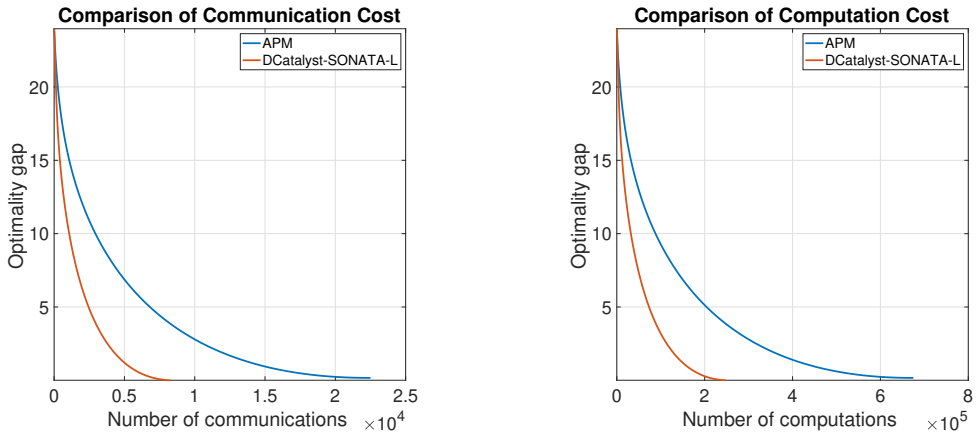

Figure 3: The comparison between APM(Li et al., 2020b) and DCatalyst-SONATA-L under convex and non-smooth setting in **(a):** communication cost (left) and **(b):** computation cost (right).

We contrast the proposed DCatalyst-SONATA-L with the APM algorithm (Li et al., 2020b), which to our knowledge is the only algorithm available in the literature applicable to such a class of problems. The tuning of the free parameters in APM follows the theoretical guidelines in (Li et al., 2020b). For the DCatalyst-SONATA-L algorithm, the parameters are chosen according to the theory developed in Sec. 3.4: we set $\delta = L$, the momentum parameter is calculated using (10), and the number of inner-loop iterations is set as $\lceil \log(k+1) \rceil$ for all k [see (12)].

In Figure 3, we plot the optimality gap achieved by the two algorithms versus the number of communications (left-panel) and computations (right-panel). The figure confirms the sublinear convergence of the algorithms, with DCatalyst-SONATA-L outperforming APM (Li et al., 2020b) both on communication and computation costs.

7.3 Minimizing finite-sum functions via variance-reduction methods

The last set of experiments concerns the application of acceleration to decentralized variance reduction methods, testing the first scheme of this kind applicable to composite functions u (whose f-part has a finite-sum structure). To offer a comprehensive comparison against existing accelerated decentralized variance reduction methods, which are available only for *smooth*, unconstrained optimization problems, we also present in Sec. 7.3.2, some experiments for smooth functions u.

7.3.1 Logistic regression with elastic net regularization

We consider the logistic regression problem introduced in Sec. 7.1 (see (42)), but now exploiting algorithmically the finite-sum structure of each agent's loos f_i, that is,

$$f_i(x) = \sum_{j=1}^{n} f_{ij}(x), \quad \text{with} \quad f_{ij}(x) = \log(1 + \exp(-b_{ij} \cdot \langle x, a_{ij} \rangle)) + \frac{\gamma}{2}\|x\|_2^2, \quad r(x) = \lambda \|x\|_1.$$

In the experiments, we set $\lambda = 10^{-7}$ and $\gamma = 10^{-4}$, and use only the first $N = 6000$ data points of the data set MNIST. This ensures $n = 200 << \kappa_s \approx 2.73 \times 10^5$.

Since for such classes of functions there is no accelerated variance reduction decentralized methods in the literature, we compare the non-accelerated decentralized algorithm PMGT-LSVRG (Ye et al., 2026) with its accelerated counterpart obtained applying our DCatalyst framework, termed DCatalyst-PMGT-LSVRG, The tuning parameters of PMGT-LSVRG are set as recommended in (Ye et al., 2026). For DCatalyst-PMGT-LSVRG, similar to our previous experiments, we set $\delta = L - \mu$, the momentum parameter $\alpha = \sqrt{\mu/(\mu+\delta)}$, and the number of inner loop iterations $T_k = \lceil \log(L/\mu) \rceil$ Figure 4 plots the optimality gap versus the number of communications and iterations (evaluated in terms of single gradient computation ∇f_{ij}) produced by the two algorithms. The plots show that, when $n << \kappa_s$, the acceleration introduced by DCatalyst improves both computation and communication complexities of the plain PMGT-LSVRG algorithm.

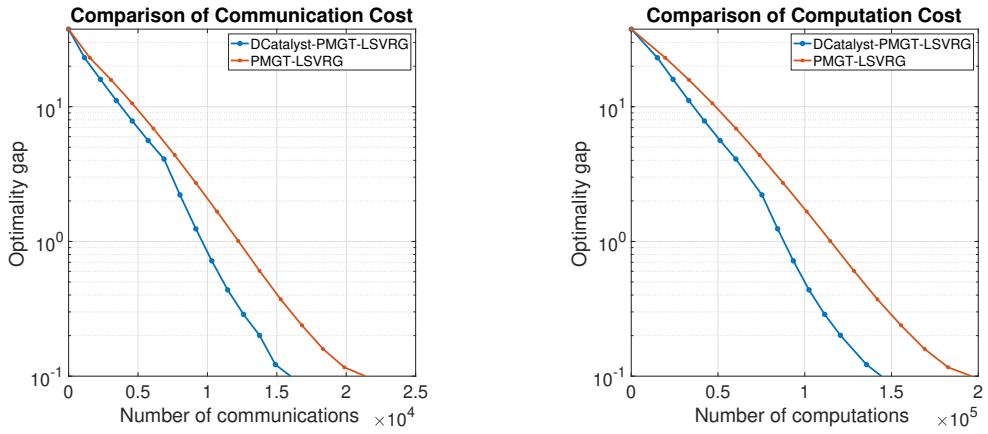

Figure 4: Comparison between PMGT-LSVRG(Ye et al., 2026) and DCatalyst-PMGT-LSVRG, under the finite-sum setting, with strongly convex non-smooth u and $n << \kappa_s$. **(a):** communication cost (left); **(b):** computation cost (right).

7.3.2 Ridge regression

As instance of strongly convex and smooth objective function with sum-structure we consider here the ridge regression model, which corresponds to Problem (P) with

$$f_i(x) = \sum_{j=1}^{n} f_{ij}(x), \quad \text{with} \quad f_{ij}(x) = \frac{1}{2}\|a_{ij}^T x - b_{ij}\|^2 + \frac{0.1}{2}\|x\|^2, \quad r(x) = 0.$$

Here, $\{(a_{i,j}, b_{ij})\}_{j=1}^n$, $a_{ij} \in \mathbb{R}^d$, $b_{ij} \in \mathbb{R}$, is the local data set accessible only to agent i.

We simulate two notable decentralized algorithms that utilize variance reduction techniques and applicable to smooth functions: the Acc-VR-EXTRA-CA algorithm and the Acc-VR-DIGing-CA algorithm (Li et al., 2022). We compare these with our DCatalyst framework applied to the (non-accelerated) VR-EXTRA (Li et al., 2022), which we term DCatalyst-VR-EXTRA. For Acc-VR-EXTRA-CA and Acc-VR-DIGing-CA, we adhere to the tuning recommendations provided in (Li et al., 2022). For DCatalyst-VR-EXTRA, similar to our previous experiments, we set $\delta = L - \mu$, the momentum parameter $\alpha = \sqrt{\mu/(\mu+\delta)}$, and the number of inner loop iterations $T_k = \lceil 0.5 \log(L/\mu) \rceil$. Additionally, to illustrate the balance between communication and computation costs, we experiment with different batch sizes: $b = 1$, $b = 100$, and $b = 1000$.

Figure 5 presents the main results. It plots the optimality gap achieved by the aforementioned algorithms as a function of the number of communication rounds (left panel) and the number of iterations (right panel), where iterations are measured as local gradient ∇f_{ij} evaluations. The results indicate that DCatalyst-VR-EXTRA surpasses both Acc-VR-EXTRA-CA and Acc-VR-DIGing-CA in terms of both computation and communication costs. Additionally, the figure highlights an interesting trade-off in DCatalyst-VR-EXTRA when varying the batch size b: as b grows, the communication cost reduces and the computation cost increases, increases, the communication cost decreases while the computation

cost increases, demonstrating the efficiency gains from adjusting batch sizes within our DCatalyst framework.

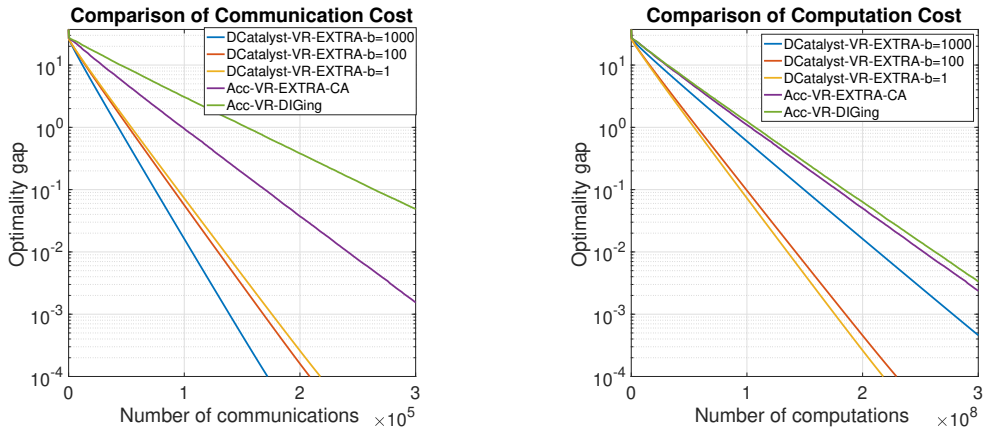

Figure 5: Comparison of distributed algorithms under finite-sum setting with a strongly convex smooth objective function in **(a):** communication cost (left) and **(b):** Computation cost (right).

8. Conclusion

We studied decentralized optimization for (strongly) convex composite objectives over general network topologies and proposed *DCatalyst*, a unified black-box framework that adds Nesterov-type acceleration to a broad class of decentralized algorithms. DCatalyst covers multiple problem classes–strongly convex and merely convex, smooth and composite, with and without finite-sum structure–and, when instantiated with existing methods such as SONATA and PMGT-LSVRG, it yields new accelerated decentralized algorithms for problem classes that previously had no accelerated solutions. Our complexity results show that, up to logarithmic factors, DCatalyst achieves optimal communication and computational guarantees, and our experiments confirm substantial practical gains, especially in terms of communication savings on ill-conditioned problems.

Analytically, our main contribution is the *inexact estimating sequences* framework, a new extension of Nesterov's estimating sequences tailored to composite decentralized optimization. By explicitly incorporating *exogenous* error sequences, this machinery accommodates multiple agent-specific iterates, consensus errors, and inexact decentralized subproblem solutions within a single black-box analysis, and naturally induces inner-loop termination rules and warm-start strategies that ensure accelerated outer-loop convergence. An interesting direction for future work is to remove the residual logarithmic factors in our rates and develop a unified acceleration framework with truly optimal complexity, potentially via a single-loop design that bypasses the outer–inner decomposition altogether.

Acknowledgments

This work has been supported by the Office of Naval Research (ONR Grant N. N000142412751).

Appendix A.

A.1 Proof of Lemma 19

Since $\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k) \leq \psi_i^{k,\star} + \epsilon_{\psi,i}^k$, we have $\psi_i^{k,\star} + \epsilon_{\psi,i}^k - \mathtt{M}^\star_{\frac{1}{\delta}u} \geq 0$. Then

$$\begin{aligned}0 &\leq \epsilon_{\psi,i}^k + \psi_i^{k,\star} - \mathtt{M}^\star_{\frac{1}{\delta}u} \leq \epsilon_{\psi,i}^k + \psi_i^k(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} \\ &\leq (1-\alpha^{k-1})(\psi_i^{k-1}(x^\star) + \epsilon_{\psi,i}^{k-1} - \mathtt{M}^\star_{\frac{1}{\delta}u}) + \epsilon_{\psi,i}^k + \alpha^{k-1}\epsilon_i^{k-1} - (1-\alpha^{k-1})\epsilon_{\psi,i}^{k-1},\end{aligned}$$

where the last inequality is due to (ii) of Definition 18. Dividing by λ^k both sides of the last inequality, yields

$$\frac{\psi_i^k(x^\star) + \epsilon_{\psi,i}^k - \mathtt{M}^\star_{\frac{1}{\delta}u}}{\lambda^k} \leq \frac{\psi_i^{k-1}(x^\star) + \epsilon_{\psi,i}^{k-1} - \mathtt{M}^\star_{\frac{1}{\delta}u}}{\lambda^{k-1}} + \frac{\epsilon_{\psi,i}^k - (1-\alpha^{k-1})\epsilon_{\psi,i}^{k-1} + \alpha^{k-1}\epsilon_i^{k-1}}{\lambda^k}. \qquad \square$$

A.2 Proof of Proposition 20

It follows from (20), $\psi_i^k(x^\star) + \epsilon_{\psi,i}^k - \mathtt{M}^\star_{\frac{1}{\delta}u} \geq \psi_i^{k,\star} + \epsilon_{\psi,i}^k - \mathtt{M}^\star_{\frac{1}{\delta}u} \geq \mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k) - \mathtt{M}^\star_{\frac{1}{\delta}u}$. We have

$$\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k) - \mathtt{M}^\star_{\frac{1}{\delta}u} \leq \psi_i^k(x^\star) + \epsilon_{\psi,i}^k - \mathtt{M}^\star_{\frac{1}{\delta}u} \leq \lambda^k \left(\psi_i^0(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} + \sum_{j=0}^{k-1} \frac{\epsilon_{tot,i}^j}{\lambda^{j+1}} \right), \qquad (43)$$

We separate the discussion into the two cases $\mu > 0$ and $\mu = 0$.
(i) $\mu > 0$**:** Summing (43) over i while taking the expected value, and using $\lambda^k = (1-\alpha)^k$, we have

$$\begin{aligned}&\frac{1}{m}\sum_{i=1}^m \lambda^k \left(\psi_i^0(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} + \sum_{j=0}^{k-1} \frac{\mathbb{E}[\epsilon_{tot,i}^j]}{\lambda^{j+1}} \right) \leq (1-c\alpha)^k \frac{1}{m}\sum_{i=1}^m \left(\psi_i^0(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} + \frac{C_{scvx}}{1-c\alpha} \sum_{j=0}^{k-1} \frac{(1-\alpha)^{k-1-j}}{(1-c\alpha)^{k-1-j}} \right) \\ &\leq (1-c\alpha)^k \frac{1}{m}\sum_{i=1}^m \left(\psi_i^0(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} + \frac{C_{scvx}}{\alpha(1-c)} \right).\end{aligned}$$

Invoking the $\mu_{\mathtt{M}}$-strong convexity of $\mathtt{M}_{\frac{1}{\delta}u}$, we deduce

$$\begin{aligned}\frac{1}{m}\mathbb{E}[\|\tilde{\mathbf{x}}^k - \mathbf{x}^\star\|^2] &\leq \frac{2}{m\mu_{\mathtt{M}}} \sum_{i=1}^m \left(\mathbb{E}[\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k)] - \mathtt{M}^\star_{\frac{1}{\delta}u} \right) \leq (1-c\alpha)^k \frac{1}{m}\sum_{i=1}^m \left(\frac{2}{\mu_{\mathtt{M}}} \left(\psi_i^0(x^\star) - \mathtt{M}^\star_{\frac{1}{\delta}u} + \frac{C_{scvx}}{\alpha(1-c)} \right) \right) \\ &= c_{scvx}\,(1-c\alpha)^k\,.\end{aligned}$$

(ii) $\mu = 0$**:** We hinge on the lemma below, which follows from (Lin et al., 2015) applied to our setting.

Lemma 24 *Let $\{\lambda^k\}_k$ be defined in (21) where $\{\alpha^k\}_k$ is defined as in Algorithm 1 for the case $\mu = 0$. Then, for all $k \geq 0$,*

$$\frac{4}{(k+2)^2} \geq \lambda^k \geq \frac{2}{(k+2)^2}.$$

Using Lemma 24, we have

$$\begin{aligned}
&\frac{1}{m}\sum_{i=1}^{m}\left(\mathbb{E}[\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k)]-\mathtt{M}^{\star}_{\frac{1}{\delta}u}\right)\le\lambda^k\frac{1}{m}\sum_{i=1}^{m}\left(\psi_i^0(x^\star)-\mathtt{M}^{\star}_{\frac{1}{\delta}u}+\sum_{j=0}^{k-1}\frac{\mathbb{E}[\epsilon_{tot,i}^j]}{\lambda^{j+1}}\right)\\
&\le\lambda^k\frac{1}{m}\sum_{i=1}^{m}\left(\psi_i^0(x^\star)-\mathtt{M}^{\star}_{\frac{1}{\delta}u}+\sum_{j=0}^{\infty}\frac{C_{cvx}(j+3)^2}{2(j+1)^{3+r_0}}\right)\\
&\le\frac{4}{(k+2)^2}\frac{1}{m}\sum_{i=1}^{m}\left(\psi_i^0(x^\star)-\mathtt{M}^{\star}_{\frac{1}{\delta}u}+\sum_{j=1}^{\infty}\frac{C_{cvx}((j+1)^2+4)}{(j+1)^{3+r_0}}++\frac{9C_{cvx}}{2}\right)\\
&\le\frac{4}{(k+2)^2}\frac{1}{m}\sum_{i=1}^{m}\left(\psi_i^0(x^\star)-\mathtt{M}^{\star}_{\frac{1}{\delta}u}+\frac{10C_{cvx}}{r_0}\right).
\end{aligned}$$

The desired result (22) follows from the above inequality and the fact

$$\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k)-\mathtt{M}^{\star}_{\frac{1}{\delta}u}\ge\frac{1}{2\delta}\|\nabla\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k)\|^2.$$

If $r\equiv 0$, $u(x)$ is L-smooth. Define $\tilde{x}_i^{k,\star}:=\tilde{x}_i^k-(1/\delta)\nabla\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k)$. Then

$$\|\nabla\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k)\|=\|\nabla u(\tilde{x}_i^{k,\star})\|\ge\|\nabla u(\tilde{x}_i^k)\|-\frac{L}{\delta}\|\nabla\mathtt{M}_{\frac{1}{\delta}u}(\tilde{x}_i^k)\|.$$

Using this lower bound in (22) yields (23). □

Appendix B.

B.1 Proof of Lemma 8

For any variables $x_1,x_0\in\mathbb{R}^d$,

$$\begin{aligned}
&u^{k+1}(x_1)-u^k(x_1)-(u^{k+1}(x_0)-u^k(x_0))\\
&=\frac{\delta}{2m}\sum_{i=1}^{m}\left(\|x_1-z_i^{k+1}\|^2-\|x_1-z_i^k\|^2-\|x_0-z_i^{k+1}\|^2+\|x_0-z_i^k\|^2\right).
\end{aligned}$$

Hence,

$$\begin{aligned}
&u^{k+1}(x_i^{k+1})-u^{k+1}(x^{k+1,\star})=u^k(x_i^{k+1})-u^k(x^{k,\star})+u^k(x^{k,\star})-u^k(x^{k+1,\star})+\delta\langle\bar{z}^k-\bar{z}^{k+1},x_i^{k+1}-x^{k+1,\star}\rangle\\
&\le u^k(x_i^{k+1})-u^k(x^{k,\star})-\frac{\delta+\mu}{2}\|x^{k,\star}-x^{k+1,\star}\|^2+\delta\langle\bar{z}^k-\bar{z}^{k+1},x_i^{k+1}-x^{k+1,\star}\rangle.
\end{aligned}$$

Then,

$$\begin{aligned}
&\frac{2}{(\mu+\delta)m}\sum_{i=1}^{m}(u^{k+1}(x_i^{k+1})-u^{k+1}(x^{k+1,\star}))\\
&\le\frac{2}{(\mu+\delta)m}\sum_{i=1}^{m}(u^k(x_i^{k+1})-u^k(x^{k,*}))-\|x^{k,\star}-x^{k+1,\star}\|^2+\frac{2\delta}{\mu+\delta}\langle\bar{z}^k-\bar{z}^{k+1},\bar{x}^{k+1}-x^{k+1,\star}\rangle.
\end{aligned}\tag{44}$$

We bound the last term on the RHS of the inequality as follows:

$$\begin{aligned}\frac{2\delta}{\mu+\delta}\langle \bar{z}^k-\bar{z}^{k+1}, x^{k,\star}-x^{k+1,\star}\rangle &\leq \|x^{k,\star}-x^{k+1,\star}\|^2+\frac{\delta^2}{(\delta+\mu)^2}\|\bar{z}^k-\bar{z}^{k+1}\|^2,\\ \frac{2\delta}{\mu+\delta}\langle \bar{z}^k-\bar{z}^{k+1}, \bar{x}^{k+1}-x^{k,\star}\rangle &\leq \|x^{k,\star}-\bar{x}^{k+1}\|^2+\frac{\delta^2}{(\delta+\mu)^2}\|\bar{z}^k-\bar{z}^{k+1}\|^2.\end{aligned} \tag{45}$$

Combine (44) and (45),

$$\begin{aligned}&\frac{2}{(\mu+\delta)m}\sum_{i=1}^m(u^{k+1}(x_i^{k+1})-u^{k+1}(x^{k+1,\star}))\\ &\leq \frac{2}{(\mu+\delta)m}\sum_{i=1}^m(u^k(x_i^{k+1})-u^k(x^{k,*}))+\frac{2\delta^2}{(\delta+\mu)^2}\|\bar{z}^k-\bar{z}^{k+1}\|^2+\|\bar{x}^{k+1}-x^{k,\star}\|^2\\ &\leq \frac{4}{(\mu+\delta)m}\sum_{i=1}^m(u^k(x_i^{k+1})-u^k(x^{k,*}))+\frac{2\delta^2}{(\delta+\mu)^2}\|\bar{z}^k-\bar{z}^{k+1}\|^2.\end{aligned} \tag{46}$$

Using the initialization (13),

$$\begin{aligned}&\frac{\eta}{m}\left(4(L_{\max}+\delta)^2\|\mathbf{x}_\perp^{k+1}\|^2+2\|\mathbf{y}_\perp^{k+1}\|^2\right)\\ &=\frac{\eta}{m}\left(4(L_{\max}+\delta)^2\|\mathbf{x}_\perp^{k,T_k}\|^2+2\|\mathbf{y}_\perp^{k,T_k}+\delta(I-\frac{1}{m}\mathbf{1}_m\mathbf{1}_m^T)(\mathbf{z}^k-\mathbf{z}^{k+1})\|^2\right)\\ &\leq\frac{\eta}{m}\left(4(L_{\max}+\delta)^2\|\mathbf{x}_\perp^{k,T_k}\|^2+4\|\mathbf{y}_\perp^{k,T_k}\|^2+16\delta^2\|\mathbf{z}^{k+1}-\mathbf{z}^k\|^2\right).\end{aligned} \tag{47}$$

Combine (46) and (47), yields

$$\begin{aligned}\mathcal{L}^{k+1}(\boldsymbol{s}^{k+1,0})&\leq 2\mathcal{L}^k(\boldsymbol{s}^{k,T_k})+\frac{2\delta^2}{(\delta+\mu)^2}\|\bar{z}^{k+1}-\bar{z}^k\|^2+\frac{16\eta\delta^2}{m}\|\mathbf{z}^{k+1}-\mathbf{z}^k\|^2\\ &\leq 2L^k(\boldsymbol{s}^{k,T_k})+\left(\frac{2\delta^2}{m(\delta+\mu)^2}+\frac{16\eta\delta^2}{m}\right)\|\mathbf{z}^{k+1}-\mathbf{z}^k\|^2. \qquad \square\end{aligned}$$

B.2 Proof of Corollary 9

By Lemma 7 and 8, SONATA algorithm (under the initialization (13)) satisfies Assumptions 4 and 5. Next, we provide a suitable value for T_k. It is trivial that check that (14) holds and $r_{\mathcal{M},\delta}=\mathcal{O}(1)$. Let us choose $\epsilon_0=\max\left\{(1/m)\|\mathbf{x}^\star-\mathbf{x}^0\|^2,\mathcal{L}^0(\boldsymbol{s}^{0,0})\right\}$. By (38), it follows

$$\begin{aligned}c_{scvx}&\leq\frac{1}{m}\left(2+\frac{\delta}{\mu}\right)\|\mathbf{x}^\star-\mathbf{x}^0\|^2+\frac{2(\mu+\delta)^2}{\mu^2(1-c)}\|\mathbf{x}^\star-\mathbf{x}^0\|^2+\left(\frac{\mu+\delta}{\mu(1-c)}+\frac{2\sqrt{2000}(\mu+\delta)^2\sqrt{m}}{\mu^2(1-c)^2}\right)\epsilon_0\\ &=\left(2+\frac{\delta}{\mu}+\frac{(2m+1)(\mu+\delta)^2}{\mu^2(1-c)}+\frac{2\sqrt{2000}(\mu+\delta)^2\sqrt{m}}{\mu^2(1-c)^2}\right)\epsilon_0,\end{aligned}$$

and, according to Proposition 2, (7) holds under the following bound on T_k:

$$r_{\mathcal{M},\delta}\log\frac{c_{\mathcal{L}}\,c_{\mathcal{M}}+36d_{\mathcal{M}}\,c_{scvx}\dfrac{1}{\epsilon_0(1-c\,\alpha)^2}}{1-c\alpha}=r_{\mathcal{M},\delta}\log\frac{2+36\left(\frac{2\delta^2}{(\delta+\mu)^2}+16\eta\delta^2\right)\dfrac{c_{scvx}}{\epsilon_0(1-c\,\alpha)^2}}{1-c\alpha}$$
$$\leq r_{\mathcal{M},\delta}\log\frac{2+\frac{36}{(1-c\,\alpha)^2}\left(\frac{2\delta^2}{(\delta+\mu)^2}+16\eta\delta^2\right)\left(2+\frac{\delta}{\mu}+\frac{(2m+1)(\mu+\delta)^2}{\mu^2(1-c)}+\frac{2\sqrt{2000}(\mu+\delta)^2\sqrt{m}}{\mu^2(1-c)^2}\right)}{1-c\alpha}.$$

We can now apply Theorem 3. The total numbers of inner-plus-outer iterations $N_{\mathtt{it}}$ of DCatalyst-SONATA-F and DCatalyst-SONATA-L are respectively

$$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\beta}{\mu}}\log\frac{1}{\epsilon}\right)\quad\text{and}\quad\widetilde{\mathcal{O}}\left(\sqrt{\kappa_g}\log\frac{1}{\epsilon}\right).$$

When solving subproblem (P^k), the network connectivity ρ is required to satisfy (Lemma 7)

$$\rho=\begin{cases}\mathcal{O}\left(\left(\dfrac{\beta}{L+\beta-\mu}+1\right)^2\right), & \text{(DCatalyst-SONATA-F);}\\ \mathcal{O}\left(\dfrac{L}{L_{\max}+L-\mu}\right), & \text{(DCatalyst-SONATA-L).}\end{cases}$$

This conditons can be eforced by running multiple communication rounds per algorithm iteration. Specifically, employing Chebyshev acceleration (Scaman et al., 2017) yields $N_{\mathtt{com},\mathcal{M}}=\widetilde{\mathcal{O}}(\sqrt{1/(1-\rho)})$. Then, the total number of communications $N_{\mathtt{com}}$ for DCatalyst-SONATA-F and DCatalyst-SONATA-L reads respectively

$$\widetilde{\mathcal{O}}\left(\sqrt{\frac{\beta}{\mu(1-\rho)}}\log\frac{1}{\epsilon}\right)\quad\text{and}\quad\widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_g}{1-\rho}}\log\frac{1}{\epsilon}\right).$$

For the sake of completeness, we provide also the computational complexities of DCatalyst-SONATA-F (the one of DCatalyst-SONATA-L coincides with $N_{\mathtt{it}}$). Suppose that, at every iteration t of SONATA-F, each agent solves its own subproblem inexactly

$$x_i^{k,t+1}=\arg\min_{x\in\mathbb{R}^d}\underbrace{f_i(x)+\frac{\beta-\mu}{2}\|x-z_i^k\|^2+\frac{\beta}{2}\|x-x_i^{k,t}\|^2+r(x)}_{u_i^{k,t}(x)},$$

employing the accelerated proximal gradient method, such that the resulting inexact solution $\hat{x}_i^{k,t+1}$ satisfies

$$\max\left\{\frac{1}{2\beta}(u_i^{k,t}(\hat{x}_i^{k,t+1})-u_i^{k,t}(x_i^{k,t+1})),\|\hat{x}_i^{k,t+1}-x_i^{k,t+1}\|^2\right\}=\mathcal{O}\left(\epsilon_0(1-c\alpha)^{k+1}\left(1-\frac{1}{r_{\mathcal{M},\delta}}\right)\right).$$

Following similar steps as those in (Tian et al., 2022, Appendix E), the number of computations for DCatalyst-SONATA-F to meet such an inexact termination is

$$\widetilde{\mathcal{O}}\left(\sqrt{\frac{L+2\beta-\mu}{\beta-\mu}}\cdot\frac{\beta}{\mu}\log^2\frac{1}{\epsilon}\right)=\widetilde{\mathcal{O}}\left(\sqrt{\frac{L+\beta}{\beta}}\cdot\frac{\beta}{\mu}\log^2\frac{1}{\epsilon}\right)\qquad\square.$$

B.3 Proof of Corollary 10

Lemma 7 and Lemma 8 hold. To found a suitable bound of T_k, we let $\epsilon_0 := \max\{B^2, \mathcal{L}^0(\boldsymbol{s}^{0,0})\}$, then

$$r_{\mathcal{M},\delta} \log\left(c_{\mathcal{L}} c_{\mathcal{M}} 2^{4+2r_0} + \frac{36 c_{\mathcal{L}} d_{\mathcal{M}} B^2 (k+3)^{4+2r_0}}{\epsilon_0}\right) \leq r_{\mathcal{M},\delta} \log\left(2^{5+2r_0} + 1224(k+3)^{4+2r_0}\right).$$

According to Proposition 5, (11) holds if $T_k \geq r_{\mathcal{M},\delta} \log\left(2^{5+2r_0} + 1224(k+3)^{4+2r_0}\right)$.

We can apply Theorem 6 and, following similar steps as in Sec. B.2, we obtain the communication and computational complexities for DCatalyst-SONATA-F and DCatalyst-SONATA-L as stated in the corollary. □

B.4 Proof of Lemma 12

When applying PUDA to solve the subproblem (P^k), denote the fixed point of $\{y_i^{k,t}\}_{t\geq 0}$ as $y^{k,\star}$ and $\mathbf{y}^{k,\star} := 1_m (y^{k,\star})^\top$. We have

$$\begin{aligned}
&\|\mathbf{x}^{k,T_k} - \mathbf{x}^{k+1,\star}\|^2 \leq 2\|\mathbf{x}^{k,T_k} - \mathbf{x}^{k,\star}\|^2 + 2\|\mathbf{x}^{k+1,\star} - \mathbf{x}^{k,\star}\|^2 \leq 2\|\mathbf{x}^{k,T_k} - \mathbf{x}^{k,\star}\|^2 + 2\|\mathbf{z}^{k+1} - \mathbf{z}^k\|^2, \\
&\|\mathbf{y}^{k,T_k} - \mathbf{y}^{k+1,\star}\|^2 \leq 2\|\mathbf{y}^{k,T_k} - \mathbf{y}^{k,\star}\|^2 + 2\|\mathbf{y}^{k,\star} - \mathbf{y}^{k+1,\star}\|^2.
\end{aligned} \tag{48}$$

Given $k \geq 0$, according to (Alghunaim et al., 2020, Lemma 1), the following holds

$$H\mathbf{x}^{k,\star} - \eta H \nabla F^k(\mathbf{x}^{k,\star}) - H^2 \mathbf{y}^{k,\star} = 0,$$

where $\nabla F^k(\mathbf{x}^{k,\star})$ denotes $[\nabla f_1^k(x^{k,\star}), \cdots, \nabla f_m^k(x^{k,\star})]^\top$.

Then

$$\begin{aligned}
&\sigma_{\min}^+(H^2)\|\mathbf{y}^{k+1,\star} - \mathbf{y}^{k,\star}\|^2 \leq \|H\mathbf{x}^{k,\star} - \eta H \nabla F^k(\mathbf{x}^{k,\star}) - H\mathbf{x}^{k+1,\star} + \eta H \nabla F^{k+1}(\mathbf{x}^{k+1,\star})\|^2 \\
&\leq \Big((1+\delta\eta)\|H(\mathbf{x}^{k,\star} - \mathbf{x}^{k+1,\star})\| + \eta\|H(\nabla F(\mathbf{x}^{k+1,\star}) - \nabla F(\mathbf{x}^{k,\star}))\| + \delta\eta\|H(\mathbf{z}^k - \mathbf{z}^{k+1})\|\Big)^2 \\
&\leq \sigma_{\max}(H^2)\left(9 + 9\delta^2\eta^2\right)\|\mathbf{z}^k - \mathbf{z}^{k+1}\|^2.
\end{aligned} \tag{49}$$

Combine (48) and (49), we obtain the desired expression of $c_{\mathcal{M}}$ and $d_{\mathcal{M}}$.

□

B.5 Proof of Corollary 13

By Lemma 11 and 12, Assumption 4 and 5 hold when using PUDA. With δ chosen as in (16), the convergence rate reads

$$r_{\mathcal{M},\delta} = \max\left\{\frac{1}{\sigma_{\min}^+(H^2)}, \frac{4(L_{\max}+\delta)}{(2-\sigma_{\max}(C))^2(\mu_{\min}+\delta)}\right\} = \frac{1}{\sigma_{\min}^+(H^2)}.$$

We can now provide a valid bound on T_k. Similar to the proof of Corollary 9, we pick $\epsilon_0 = \max\{(1/m)\|\mathbf{x}^\star - \mathbf{x}^0\|^2, \mathcal{L}^0(\boldsymbol{s}^{0,0})\}$, yielding

$$c_{scvx} \leq \left(2 + \frac{\delta}{\mu_{\min}} + \frac{(2m+1)(\mu_{\min}+\delta)^2}{\mu_{\min}^2(1-c)} + \frac{2\sqrt{2000}(\mu_{\min}+\delta)^2\sqrt{m}}{\mu_{\min}^2(1-c)^2}\right)\epsilon_0,$$

and thus the following bound can be used as valid value of T_k:

$$r_{\mathcal{M},\delta} \log \frac{c_{\mathcal{L}}\, c_{\mathcal{M}} + 36 d_{\mathcal{M}}\, c_{scvx} \dfrac{1}{\epsilon_0 (1 - c\,\alpha)^2}}{1 - c\alpha}$$
$$\leq r_{\mathcal{M},\delta} \log \frac{2 + \frac{36}{(1-c\,\alpha)^2}\left(2 + \frac{\sigma_{\max}(H^2)\left(9 + 9\delta^2\eta^2\right)}{\sigma^+_{\min}(H^2)}\right)\left(2 + \frac{\delta}{\mu_{\min}} + \frac{(2m+1)(\mu_{\min}+\delta)^2}{\mu^2_{\min}(1-c)} + \frac{2\sqrt{2000}(\mu_{\min}+\delta)^2\sqrt{m}}{\mu^2_{\min}(1-c)^2}\right)}{1 - c\alpha}.$$

Under the above setting, we can call Theorem 3 and obtain the total number $N_{\mathtt{it}}$ of inner-plus-outer iterations of PUDA as given in Corollary 13. According to (15), both communication and computational complexities are of the order of $N_{\mathtt{it}}$. □

B.6 Proof of Lemma 16

Denote the smoothness parameter for f^k of (S.1) in Algorithm 1 as L_δ, p_{ij} in (17) becomes $L_{\delta,ij}/(\sum_{j=1}^n L_{\delta,ij})$. We have

$$\begin{aligned}
&\|\bar{x}^{k,T_k} - x^{k+1,\star}\|^2 \leq 2\|\bar{x}^{k,T_k} - x^{k,\star}\|^2 + \frac{2}{m}\|\mathbf{z}^k - \mathbf{z}^{k+1}\|^2, \\
&\frac{1}{m}\|\mathbf{x}_\perp^{k+1}\|^2 + \frac{\eta^2}{m}\|\mathbf{y}_\perp^{k+1}\|^2 \leq \frac{1}{m}\left(\|\mathbf{x}_\perp^{k,T_k}\|^2 + 2\eta^2\|\mathbf{y}_\perp^{k,T_k}\|^2 + 8\eta^2\delta^2\|\mathbf{z}^k - \mathbf{z}^{k+1}\|^2\right),
\end{aligned} \tag{50}$$

and

$$\begin{aligned}
\Delta_f^{k+1,0} &\leq \frac{1}{mn}\sum_{i=1}^m\sum_{j=1}^n \frac{1}{np_{ij}}\left(2\|\nabla f_{ij}^{k+1}(v_i^{k,T_k}) - \nabla f_{ij}^{k+1}(x^{k,\star})\|^2 + 2\|\nabla f_{ij}^{k+1}(x^{k+1,\star}) - \nabla f_{ij}^{k+1}(x^{k,\star})\|^2\right) \\
&\leq 2\Delta_f^{k,T_k} + \frac{2\bar{L}_{\delta,\max}}{(mn)^2}\sum_{i=1}^m\sum_{j=1}^n L_{\delta,ij}\|\mathbf{z}^k - \mathbf{z}^{k+1}\|^2,
\end{aligned} \tag{51}$$

where $L_{\delta,ij} = L_{ij} + \delta$ and $\bar{L}_{\delta,\max} = \bar{L}_{\max} + \delta$. The proof of the lemma follows readily combining (50) and (51). □

B.7 Proof of Corollary 17

By Lemma 15 and 16, PMGT-LSVRG (under the initialization (18)) satisfies Assumption 4 and 5. The convergence rate is $r_{\mathcal{M},\delta} = 48n$.

We proceed bounding T_k. Similarly to the procedure in the analysis of Corollary 9, we set $\epsilon_0 = \max\{(1/m)\left\|\mathbf{x}^0 - \mathbf{x}^\star\right\|^2, \mathcal{L}^0(\boldsymbol{s}^{0,0})\}$. Then,

$$c_{scvx} \leq \left(2 + \frac{\delta}{\mu} + \frac{(2m+1)(\mu+\delta)^2}{\mu^2(1-c)} + \frac{2\sqrt{2000}(\mu+\delta)^2\sqrt{m}}{\mu^2(1-c)^2}\right)\epsilon_0.$$

Invoking Proposition 2, the following bound holds for T_k:

$$r_{\mathcal{M},\delta} \log \frac{2 + 36\left(2 + 8\eta^2\delta^2 + \frac{8\eta^2(\bar{L}_{\max}+\delta)^2}{n^2}\right)\frac{c_{scvx}}{\epsilon_0(1-c\,\alpha)^2}}{1-c\alpha}$$
$$\leq r_{\mathcal{M},\delta} \log \frac{2 + \frac{36}{(1-c\,\alpha)^2}\left(2 + 8\eta^2\delta^2 + \frac{8\eta^2(\bar{L}_{\max}+\delta)^2}{n^2}\right)\left(2 + \frac{\delta}{\mu} + \frac{(2m+1)(\mu+\delta)^2}{\mu^2(1-c)} + \frac{2\sqrt{2000}(\mu+\delta)^2\sqrt{m}}{\mu^2(1-c)^2}\right)}{1-c\alpha}$$
$$\leq r_{\mathcal{M},\delta} \log \frac{2 + \frac{90}{(1-c\,\alpha)^2}\left(2 + \frac{\delta}{\mu} + \frac{(2m+1)(\mu+\delta)^2}{\mu^2(1-c)} + \frac{2\sqrt{2000}(\mu+\delta)^2\sqrt{m}}{\mu^2(1-c)^2}\right)}{1-c\alpha}.$$

The total number of inner-plus-outer iterations of DCatalyst-PMGT-LSVRG is then

$$N_{\texttt{it}} = \widetilde{\mathcal{O}}\left(\sqrt{\kappa_s n}\log\frac{1}{\epsilon}\right).$$

For each algorithm iteration, there are $N_{\texttt{com},\mathcal{M}} = 2N_{\texttt{FM}} = \mathcal{O}\left((1/\sqrt{1-\rho})\cdot\log n\right)$ numbers of communications. Hence, the total number of communication is

$$N_{\texttt{com}} = N_{\texttt{it}}N_{\texttt{com},\mathcal{M}} = \widetilde{\mathcal{O}}\left(\sqrt{\frac{\kappa_s n}{1-\rho}}\log\frac{1}{\epsilon}\right).$$

The computation of full-batch gradient ∇f_i requires computing all $(\nabla f_{ij})_{j\in[n]}$. In addition, at each iteration, one sample j_i is randomly selected and ∇f_{ij_i} is computed. Therefore, the total number of computations is $2N_{\texttt{it}}$ in expectation. □

B.8 Additional numerical results

We complement the main section of numerical results, with additional experiments. Specifically, we include numerical results for smooth instances of Problem (P) ($r \equiv 0$) with strongly convex or (non strongly) convex f_i. This offers a comprehensive comparison against most representative existing accelerated decentralized methods on smooth instances of Problem (P) while further corroborating our theoretical results in Sec. 3.3 and Sec. 3.4.

B.8.1 ℓ_2-regularized Logistic Regression

As a strongly convex and smooth instance of Problem (P) ($r \equiv 0$), we consider the decentralized logistic regression model with ℓ_2 regularization, which corresponds to the formulation

$$f_i(x) = \frac{1}{n}\sum_{j=1}^{n}\log(1+\exp(-b_{ij}\cdot\langle x, a_{ij}\rangle)) + \frac{\gamma_i}{2}\|x\|_2^2, \qquad r(x) = 0,$$

where $a_{ij} \in \mathbb{R}^d$ and $b_{ij} \in \{-1, 1\}$. Here, the data set $\{(a_{ij}, b_{ij})\}_{j=1}^n$ is assumed to be private and owned only by agent i. We use MNIST dataset from LIBSVM (Chang and Lin, 2011) of size $N = 60000$ and feature dimension $d = 784$. To control the values of κ_g and κ_ℓ, we set each $\gamma_i = 0.05$, for $i = 1, 2, \cdots 29$ and $\gamma_{30} = 0.005$. Under this setting, we have $\kappa_g \approx 10.8494 << \kappa_l \approx 2318.6468$ and $\beta \approx 1.7035 < L \approx 10.4896$.

We compare the proposed DCatalyst-SONATA-L and DCatalyst-SONATA-F with the representative existing accelerated decentralized algorithms, namely: Mudag (Ye et al.,

2023), and OPAPC (Kovalev et al., 2020). All algorithms were implemented according to their theoretical guidelines (Ye et al., 2023; Kovalev et al., 2020). For DCatalyst-SONATA-L (resp. DCatalyst-SONATA-F), we set $\delta = L - \mu$ (resp. $\delta = \beta - \mu$), the momentum parameter $\alpha = \mu/(\mu+\delta)$ (resp. $\alpha = \mu/(\mu+\delta)$), and the number of inner-loop iterations $T_k = \lceil \log(L/\mu) \rceil$ (resp. $T_k = \lceil \log(\beta/\mu) \rceil$), for all k. In DCatalyst-SONATA-L, the local agents' prox-updates are computed in closed-form while in DCatalyst-SONATA-F, the accelerated proximal gradient method is employed, rub up to a tolerance of 10^{-8}.

Figure 6 summarizes the comparison, plotting the optimality gap versus the number of total communications (left panel) and the number of inner-plus-outer iterations (right panel). For DCatalyst-SONATA-F, this include also the number of iterations run by the accelerated proximal gradient method employed locally to compute the local agents' prox-updates. The figure confirms linear convergence of the algorithms. DCatalyst-SONATA-F is the only accelerated algorithm exploiting function similarity, yielding much less communications, at the cost of more (albeit limited) local computations. Notably, both DCatalyst-SONATA-L and DCatalyst-SONATA-F outperform OPAPC in communication and computation efficiency. This is because their convergence depends on the global condition number κ_g rather than the local one κ_l. DCatalyst-SONATA-L outperforms Mudag in term of communications but requires more local computations. This is compatible with our theoretical findings, predicting an extra poly-log factor in the convergence of DCatalyst-SONATA-L with respect to the complexity of Mudag.

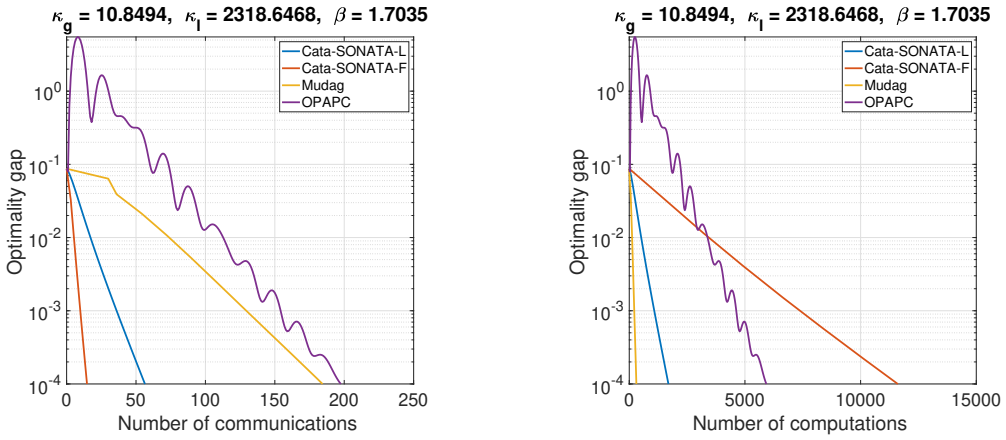

Figure 6: Comparison of distributed algorithms under strongly convex and smooth setting in **(a):** communication cost (left) and **(b):** computation cost (right).

B.8.2 Linear Regression ($N < d$) with Huber loss regularization

As (non strongly) convex, smooth instance of Problem (P) ($r \equiv 0$), we consider the underdetermined linear regression model, with Huber loss regularization (Huber and Ronchetti, 2011). Proposed for robust statistical procedures, the Huber norm has numerous applica-

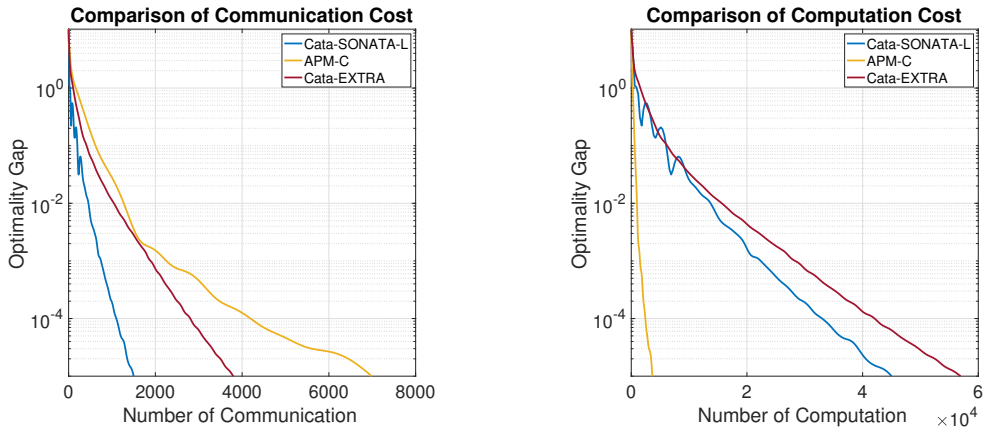

Figure 7: Comparison of distributed algorithms under convex and smooth setting in **(a):** communication cost (left) and **(b):** computation cost (right).

tions in statistics and engineering-see, e.g., (Zoubir et al., 2018). Specifically,

$$f_i(x) = \sum_{j=1}^{n}(a_{ij}^\top x - b_{ij})^2 + \sum_{k=1}^{d} r_H^i(x_k), \qquad r(x) = 0,$$

where x_k denotes the $k-$th element of $x \in \mathbb{R}^d$ and the Huber norm is given by

$$r_H^i(x_k) = \begin{cases} \lambda_i(|x_k| - \dfrac{\lambda_i}{4\gamma_i}), & |x_k| \geq \dfrac{\lambda_i}{2\gamma_i}, \\ \gamma_i x_k^2, & |x_k| < \dfrac{\lambda_i}{2\gamma_i}. \end{cases}$$

Note that $a_{ij} \in \mathbb{R}^d$ and $b_{ij} \in \{-1, 1\}$. Here, the data set $\{(a_{ij}, b_{ij})\}_{j=1}^n$ is assumed to be private and owned only by agent i. We use the MNIST from LIBSVM (Chang and Lin, 2011) as dataset, and pick only $N = 600$ data with feature dimension $d = 784$, so that $N < d$. In the experiment, we set $\lambda_i = 0.5$ for all i. We set $\gamma_i = 0.5$ for all i except $i = 30$ which is set to be 50 so that $L_{\max} \approx 137.72 > L \approx 39.78$ and a significant difference between parameters $L_{\max}$ and L is ensured.

We compare the proposed DCatalyst-SONATA-L with the APM-C (Li et al., 2020b) and Cata-EXTRA (Li and Lin, 2020). All the parameters are tuned according to their theoretical recommended values (Li et al., 2020b; Li and Lin, 2020). For the DCatalyst-SONATA-L algorithm, we set $\delta = L$, the momentum parameter is calculated using (10), and the number of inner-loop iterations is set as $\lceil \log(k+1) \rceil$, for all k [see (12)].

In Figure 7, we plot the optimality gap achieved by the above algorithms versus the number of communications (left panel) and computations (right panel) respectively. The figure confirms the sublinear convergence of the algorithms. We notice that DCatalyst-SONATA-L outperforms all other algorithms in term of communication. As predicted by our theory, DCatalyst-SONATA-L loses a bit to the APM-C in term of computations; this

is due to an extra poly-log factor and $\log \frac{1}{\epsilon}$ term appearing in its complexity. This gap reduces when $L_{\max}$ grows with respect to L.

Appendix C.

In this section, we summarize the notation used in the paper.

Table 4: Universal constants independent of iterations.

Notation	**Meaning / reference**
(P)	original optimization problem
m	# of agents in the network
d	dimension of decision variable x
d'	dimension of auxiliary variable y in the nonaccelerated algorithm $\mathcal{M}$
$\mathbf{x}^\star = \mathbf{1}_m (x^\star)^\top$	optimal solution
$u^\star$	optimal objective value
$\mathcal{D}$	distribution of samples in the finite-sum setting
ξ_{ij}	j-th sample on agent i (finite-sum)
n	# of samples per agent (finite-sum)
β	similarity coefficient
ρ	network connectivity
ϵ	accuracy parameter in rate expressions
L_{ij}	Lipschitz constant of ∇f_{ij}
L_i	Lipschitz constant of ∇f_i
μ_i	strong convexity modulus of f_i
$L_{\max}$	$\max_{i\in[m]} L_i$
$\mu_{\min}$	$\min_{i\in[m]} \mu_i$
$\overline{L_i}$	$\frac{1}{n}\sum_{j=1}^n L_{ij}$
L	Lipschitz constant of ∇f
μ	strong convexity modulus of f
$\overline{L}_{\max}$	$\max_{i\in[m]} L_i$
κ_ℓ	$L_{\max}/\mu_{\min}$
κ_g	L/μ
$\widetilde{\kappa}_s$	$\max_{i,j} L_{ij}/\mu_{\min}$
κ_s	$\overline{L}_{\max}/\mu_{\min}$
$\mathtt{M}^*_{\eta r}$	minimum of the Moreau envelope $\mathtt{M}_{\eta r}(x)$
η	stepsize of inner algorithm
α	extrapolation coefficient (used in analysis)

(table continues)

Notation	Meaning / reference
$c_{\mathcal{L}}, r_{\mathcal{M}}$	algorithm-dependent constants (Assumption 4)
δ	quadratic coefficient
$c, c_{\text{scvx}}, \epsilon_0$	constants (Proposition 1)
$c_{\mathcal{M}}, d_{\mathcal{M}}$	algorithm-dependent constants (Assumption 5)
r_0, c_{cvx}	universal constants (Proposition 4)
B	uniform bound of $\|x_i^k\|$ (Proposition 5)
$H, C \in \mathbb{R}^{m\times m}$	undetermined matrices in PUDA
$\mu_{\mathcal{M}}$	strong convexity of $\mathtt{M}_{\frac{1}{\delta}u}$
$N_{\mathtt{FM}}$	# communications per iteration in PMGT-LSVRG
p_{ij}	sampling probability of the j-th sample on agent i in PMGT-LSVRG
$C_{\text{scvx}}, C_{\text{cvx}}$	problem-dependent constants (Proposition 20)
$(\Omega, \mathcal{F}, \mathbb{P})$	probability space (stochastic setting)
$c_{\mathtt{pm}}, r_{\mathtt{pm}}, \rho_{\mathtt{pm}}$	universal constants (Lemma 15)

Table 5: Functions/operators independent of iterations.

Notation	Meaning / reference
$\ell(\cdot, \xi)$	loss at sample ξ
$f_{ij}(x)$	$\ell(x, \xi_{ij})$
$f_i(x)$	$\frac{1}{n}\sum_{j=1}^n f_{ij}(x)$
$u(x) = f(x) + r(x)$	composite objective, $f(x) = \frac{1}{m}\sum_{i=1}^m f_i(x)$
$\sigma_{\max},\ \sigma_{\min}^+$	maximum singular value and minimum positive singular value
$\mathtt{M}_{\eta r}(x)$	Moreau envelope (stepsize η, function r)
$\mathtt{prox}_{\eta r}$	proximal operator (stepsize η, function r)
$\mathcal{M}$	operator of the unaccelerated algorithm
$\mathcal{L}$	merit function associated with $\mathcal{M}$
$\approx_{\mathcal{M}} \arg\min$	solve with $\mathcal{M}$ up to some precision
$\nabla F(\mathbf{x})$	$[\nabla f_1(x_1), \ldots, \nabla f_m(x_m)]^\top$

Table 6: Iterates and iteration-dependent constants.

Notation	Meaning / reference
(P^k)	subproblem at outer iteration k
$x_i^{k,t} \in \mathbb{R}^d$	iterate of agent i at inner it. t / outer it. k
$y_i^{k,t} \in \mathbb{R}^{d'}$	auxiliary iterate at inner t / outer k
$z_i^k \in \mathbb{R}^d$	extrapolation iterate at outer iteration k
$\mathbf{x}^{k,\star} = \mathbf{1}_m (x^{k,\star})^\top$	minimizer of (P^k)
$\mathbf{y}^{k,*}$	limit point of auxiliary variables using $\mathcal{M}$ in (P^k)
$s_i^{k,t}$	$(x_i^{k,t}, y_i^{k,t})$
$r_{\mathcal{M},\delta}$	counterpart of $r_{\mathcal{M}}$ in (P^k)
α^k	extrapolation constant at outer iteration k; also used in Definition 18 / Lemma 21 (inexact estimating sequence)
T_k	number of inner iterations at outer iteration k
$N_{\mathtt{it}}$	total iterations up to outer K: $\sum_{k=0}^K T_k$
$N_{\mathtt{com},\mathcal{M}}$	# communications per $\mathcal{M}$-iteration
$N_{\mathtt{com}}$	$N_{\mathtt{com},\mathcal{M}} \cdot N_{\mathtt{it}}$
$\psi_i^k,\ \epsilon_i^k$	terms in the estimating sequence (Def. 18)
$\lambda^k,\ \epsilon_{tot,i}^k,\ \widetilde{x}_i^k$	sequences in Lemma 19
$\widetilde{z}_i^k,\ e_i^k$	sequences in Lemma 21
$\psi_i^{k,*},\ \zeta^k,\ v_i^k$	sequences in Lemma 22 (canonical form)
c_T	constants depending on $r_{\mathcal{M},\delta}$ (Prop. 2 proof)
c_{T_k}	constants depending on $r_{\mathcal{M},\delta}$ and k (Prop. 5 proof)
L_δ	Lipschitz constant of ∇f^k (Lemma 16)
Ω^k	$\prod_{t=0}^{T_k} \mathbb{R}^{m\times(d+d')}$
$\mathcal{B}_k$	Borel σ-algebra of $\prod_{\ell=0}^k \Omega^\ell$
P_k	Borel probability measures induced by $\mathcal{M}$ on $\mathcal{B}_k$
$\mathcal{F}_{k+1}$	$\sigma\Big(\{\{\boldsymbol{s}^{\ell,t}\}_{t=0}^{T_\ell}\}_{\ell=0}^k\Big)$
ω	a realization of $\{(\mathbf{x}^{k,t}, \mathbf{y}^{k,t})\}_{t=0}^{T_k}$

Table 7: Functions/operators dependent on iterations.

Notation	Meaning / reference
$f_i^k(x)$	$f_i(x) + \frac{\delta}{2}\|x - z_i^k\|^2$
$f^k(x)$	$\frac{1}{m}\sum_{i=1}^m f_i^k(x)$
$u^k(x)$	$f^k(x) + r(x)$
$\widetilde{f}_i(x; x_i^{k,t})$	local surrogate of f_i at inner t / outer k (e.g., SONATA)
$\psi_i^k(x)$	functions for canonical form of inexact estimating sequences (Lemma 22)

References

Sulaiman A Alghunaim, Ernest K Ryu, Kun Yuan, and Ali H Sayed. Decentralized proximal gradient algorithms with linear convergence rates. *IEEE Transactions on Automatic Control*, 66(6):2787–2794, 2020.

Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. *Journal of Machine Learning Research*, 18(221):1–51, 2018.

Yossi Arjevani and Ohad Shamir. Communication complexity of distributed convex learning and optimization. In *Advances in Neural Information Processing Systems*, volume 28, pages 1756–1764. Curran Associates, Inc., 2015.

Michel Baes. Estimate sequence methods: extensions and approximations. *IFOR Internal report, ETH Zurich, Switzerland*, 2009.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Ron Bekkerman, Mikhail Bilenko, and John Langford. *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Intelligent Systems and Technology*, 2(3):1–27, 2011.

Alexandre d'Aspremont, Damien Scieur, and Adrien Taylor. Acceleration methods. *Foundations and Trends in Optimization*, 5(1–2):1–245, 2021.

Paolo Di Lorenzo and Gesualdo Scutari. Next: In-network nonconvex optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 2(2):120–136, 2016.

Derek Driggs, Matthias J Ehrhardt, and Carola-Bibiane Schönlieb. Accelerating variance-reduced stochastic gradient methods. *Mathematical Programming*, 191(2):671–715, 2022.

Darina Dvinskikh and Alexander Gasnikov. Decentralized and parallel primal and dual accelerated methods for stochastic convex programming problems. *Journal of Inverse and Ill-posed Problems*, 29(3):385–405, 2021.

Pavel Dvurechensky, Dmitry Kamzolov, Aleksandr Lukashevich, Soomin Lee, Erik Ordentlich, César A Uribe, and Alexander Gasnikov. Hyperfast second-order local solvers for efficient statistically preconditioned distributed optimization. *EURO Journal on Computational Optimization*, 10:100045, 2022.

Jianqing Fan, Yongyi Guo, and Kaizheng Wang. Communication-efficient accurate statistical estimation. *Journal of the American Statistical Association*, 118(542):1000–1010, 2023.

Osman Güler. New proximal point algorithms for convex minimization. *SIAM Journal on Optimization*, 2(4):649–664, 1992.

Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. Dual-free stochastic decentralized optimization with variance reduction. In *Advances in Neural Information Processing Systems*, volume 33, pages 19455–19466. Curran Associates, Inc., 2020a.

Hadrien Hendrikx, Lin Xiao, Sebastien Bubeck, Francis Bach, and Laurent Massoulie. Statistically preconditioned accelerated gradient method for distributed optimization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 4203–4227. PMLR, 2020b.

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

Peter J Huber and Elvezio M Ronchetti. *Robust statistics*. John Wiley & Sons, 2011.

Achim Klenke. *Probability theory: a comprehensive course*. Springer, 2008.

Dmitry Kovalev, Adil Salim, and Peter Richtarik. Optimal and practical algorithms for smooth and strongly convex decentralized optimization. In *Advances in Neural Information Processing Systems*, volume 33, pages 18342–18352. Curran Associates, Inc., 2020.

Dmitry Kovalev, Egor Shulgin, Peter Richtarik, Alexander V Rogozin, and Alexander Gasnikov. Adom: Accelerated decentralized optimization method for time-varying networks. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 5784–5793. PMLR, 2021.

Guanghui Lan, Zhize Li, and Yi Zhou. A unified variance-reduced accelerated gradient method for convex optimization. In *Advances in Neural Information Processing Systems*, volume 32, pages 10462–10472. Curran Associates, Inc., 2019.

Boyue Li, Shicong Cen, Yuxin Chen, and Yuejie Chi. Communication-efficient distributed optimization in networks with gradient tracking and variance reduction. *Journal of Machine Learning Research*, 21(180):1–51, 2020a.

Huan Li and Zhouchen Lin. Revisiting extra for smooth distributed optimization. *SIAM Journal on Optimization*, 30(3):1795–1821, 2020.

Huan Li, Cong Fang, Wotao Yin, and Zhouchen Lin. Decentralized accelerated gradient methods with increasing penalty parameters. *IEEE Transactions on Signal Processing*, 68:4855–4870, 2020b.

Huan Li, Zhouchen Lin, and Yongchun Fang. Variance reduced extra and diging and their optimal acceleration for strongly convex decentralized optimization. *Journal of Machine Learning Research*, 23:1–41, 2022.

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems*, volume 30, pages 5336–5346. Curran Associates, Inc., 2017.

Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems*, volume 28, pages 3384–3392, 2015.

Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. Catalyst acceleration for first-order convex optimization: from theory to practice. *Journal of Machine Learning Research*, 18 (1):7854–7907, 2018.

Qing Ling, Wei Shi, Gang Wu, and Alejandro Ribeiro. Dlm: Decentralized linearized alternating direction method of multipliers. *IEEE Transactions on Signal Processing*, 63 (15):4051–4064, 2015.

Ji Liu and A Stephen Morse. Accelerated linear iterations for distributed averaging. *Annual Reviews in Control*, 35(2):160–165, 2011.

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

Angelia Nedić, Alex Olshevsky, and Michael G. Rabbat. Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106:953–976, 2018.

Yurii Nesterov. A method of solving a convex programming problem with convergence rate $o(1/k^2)$. *Doklady Akademii Nauk*, 269(3):543–547, 1983.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Guannan Qu and Na Li. Harnessing smoothness to accelerate distributed optimization. *IEEE Transactions on Control of Network Systems*, 5(3):1245–1260, 2017.

Sashank J Reddi, Jakub Konečnỳ, Peter Richtárik, Barnabás Póczós, and Alex Smola. Aide: Fast and communication efficient distributed optimization. *arXiv preprint arXiv:1608.06879*, 2016.

Alexander Rogozin, Vladislav Lukoshkin, Alexander Gasnikov, Dmitry Kovalev, and Egor Shulgin. Towards accelerated rates for distributed optimization over time-varying networks. In *International Conference on Optimization and Applications*, pages 258–272. Springer, 2021.

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

Mark Schmidt, Nicolas Roux, and Francis Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in Neural Information Processing Systems*, volume 24, pages 1458–1466. Curran Associates, Inc., 2011.

Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32, pages 1000–1008. PMLR, 2014.

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. *Mathematical Programming*, 207(1):1–53, 2024.

Ying Sun, Gesualdo Scutari, and Amir Daneshmand. Distributed optimization based on gradient tracking revisited: Enhancing convergence rate via surrogation. *SIAM Journal on Optimization*, 32(2):354–385, 2022.

J Kenneth Tay, Balasubramanian Narasimhan, and Trevor Hastie. Elastic net regularization paths for all generalized linear models. *Journal of Statistical Software*, 106:1–31, 2023.

Ye Tian, Gesualdo Scutari, Tianyu Cao, and Alexander Gasnikov. Acceleration in distributed optimization under similarity. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151, pages 5721–5756. PMLR, 2022.

César A. Uribe, Soomin Lee, Alexander Gasnikov, and Angelia Nedić. A dual approach for optimal algorithms in distributed optimization over networks. In *2020 Information Theory and Applications Workshop (ITA)*, pages 1–37, 2020.

Lin Xiao, Stephen Boyd, and Lall Sanjay. A scheme for robust distributed sensor fusion based on average consensus. In *4th International Symposium on Information Processing in Sensor Networks, 2005.*, pages 63–70. IEEE, 2005.

Haishan Ye, Ziang Zhou, Luo Luo, and Tong Zhang. Decentralized accelerated proximal gradient descent. In *Advances in Neural Information Processing Systems*, volume 33, pages 18308–18317. Curran Associates, Inc., 2020.

Haishan Ye, Luo Luo, Ziang Zhou, and Tong Zhang. Multi-consensus decentralized accelerated gradient descent. *Journal of Machine Learning Research*, 24(306):1–50, 2023.

Haishan Ye, Wei Xiong, and Tong Zhang. Pmgt-vr: A decentralized proximal-gradient algorithmic framework with variance reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(1):408–420, 2026.

Kun Yuan, Bicheng Ying, Xiaochuan Zhao, and Ali H. Sayed. Exact diffusion for distributed optimization and learning—part i: Algorithm development. *IEEE Transactions on Signal Processing*, 67(3):708–723, 2019.

Xiao-Tong Yuan and Ping Li. On convergence of distributed approximate newton methods: Globalization, sharper bounds and beyond. *Journal of Machine Learning Research*, 21 (206):1–51, 2020.

Yuchen Zhang and Xiao Lin. Disco: Distributed optimization for self-concordant empirical loss. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 362–370. PMLR, 2015.

Yuchen Zhang and Lin Xiao. Communication-efficient distributed optimization of self-concordant empirical loss. In *Large-Scale and Distributed Optimization*, pages 289–341. Springer, 2018.

Kaiwen Zhou, Fanhua Shang, and James Cheng. A simple stochastic variance reduced algorithm with fast convergence rates. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 5980–5989. PMLR, 2018.

Kaiwen Zhou, Qinghua Ding, Fanhua Shang, James Cheng, Danli Li, and Zhi-Quan Luo. Direct acceleration of saga using sampled negative momentum. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, volume 89, pages 1602–1610. PMLR, 2019.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2):301–320, 2005.

Abdelhak M Zoubir, Visa Koivunen, Esa Ollila, and Michael Muma. *Robust statistics for signal processing*. Cambridge University Press, 2018.