# Distributed Non-Convex First-Order Optimization and Information Processing: Lower Complexity Bounds and Rate Optimal Algorithms

Haoran Sun and Mingyi Hong [*]

September 17, 2019

### Abstract

We consider a class of popular distributed non-convex optimization problems, in which agents connected by a network $\mathcal{G}$ collectively optimize a sum of smooth (possibly non-convex) local objective functions. We address the following question: if the agents can only access the gradients of local functions, what are the *fastest* rates that any distributed algorithms can achieve, and how to achieve those rates.

First, we show that there exist difficult problem instances, such that it takes a class of distributed first-order methods at least $\mathcal{O}(1/\sqrt{\xi(\mathcal{G})} \times \bar{L}/\epsilon)$ communication rounds to achieve certain $\epsilon$-solution [where $\xi(\mathcal{G})$ denotes the spectral gap of the graph Laplacian matrix, and $\bar{L}$ is some Lipschitz constant]. Second, we propose (near) optimal methods whose rates match the developed lower rate bound (up to a ploylog factor). The key in the algorithm design is to properly embed the classical polynomial filtering techniques into modern first-order algorithms. To the best of our knowledge, this is the first time that lower rate bounds and optimal methods have been developed for distributed non-convex optimization problems.

**Keywords.** Non-convex distributed optimization; Optimal methods; Lower complexity bounds.

## 1 Introduction

### 1.1 Problem and motivation

In this work, we consider the following distributed optimization problem over a network

$$\min_{y \in \mathbb{R}^S} \ \bar{f}(y) := \frac{1}{M} \sum_{i=1}^{M} f_i(y), \tag{1}$$

where $f_i(y) : \mathbb{R}^S \to \mathbb{R}$ is a smooth and possibly non-convex function accessible to agent $i$. There is no central controller, and the $M$ agents are connected by a network defined by an *undirected* and *unweighted* graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, with $|\mathcal{V}| = M$ vertices and $|\mathcal{E}| = E$ edges. Each agent $i$ can only communicate with its immediate neighbors, and it can access one component function $f_i$ (by "access" we meant that it will be able to query the function and obtain its values and gradients; this notion will be defined precisely shortly).

---

[*]H. Sun and M. Hong are with the Department of Electrical and Computer Engineering (ECE), University of Minnesota, Minneapolis, MN 55414, USA. Email: {sun00111,mhong}@umn.edu

A common way to reformulate problem (1) in the distributed setting is given below. Introduce $M$ local variables $x_1, \cdots, x_M \in \mathbb{R}^S$ and a concatenation of $M$ variables $x := [x_1; \cdots ; x_M] \in \mathbb{R}^{SM \times 1}$, and suppose the graph $\{\mathcal{V}, \mathcal{E}\}$ is connected, then the following formulation is equivalent to the global consensus problem

$$\min_{x \in \mathbb{R}^{SM}} \ f(x) := \frac{1}{M} \sum_{i=1}^{M} f_i(x_i), \quad \text{s.t. } x_i = x_j, \forall \ (i,j) \in \mathcal{E}. \tag{2}$$

The main benefit of the above formulation is that the objective function is now separable, and the linear constraint encodes the network connectivity pattern.

## 1.2  Distributed non-convex optimization

Distributed non-convex optimization has gained considerable attention recently. For example, it finds applications in training neural networks [1], clustering [2], and dictionary learning [3], just to name a few.

The problem (1) and (2) have been studied extensively in the literature when $f_i$'s are all convex; see for example [4–6]. Primal based methods such as distributed subgradient (DSG) method [4], the EXTRA method [6], as well as primal-dual based methods such as distributed augmented Lagrangian method [7], Alternating Direction Method of Multipliers (ADMM) [8,9] have been proposed.

On the contrary, only recently there have been works addressing the more challenging problems without assuming convexity of $f_i$; see [1,3,10–23]. The convergence behavior of the distributed consensus problem (1) has been studied in [3,10,11]. Reference [12] develops a non-convex ADMM based methods for solving the distributed consensus problem (1). However the network considered therein is a star network in which the local nodes are all connected to a central controller. References [14,15] propose a primal-dual based method for unconstrained problem over a connected network, and derives a global convergence rate for this setting. In [13,17,18], the authors utilize certain gradient tracking idea to solve a constrained nonsmooth distributed problem over possibly time-varying networks. The work [19] summarizes a number of recent progress in extending the DSG-based methods for non-convex problems. References [1,16,20] develop methods for distributed stochastic zeroth and/or first-order non-convex optimization. It is worth noting that the distributed algorithms proposed in all these works converge to first-order stationary solutions, which contain local maximum, local minimum and saddle points.

Recently, the authors of [22,24–26] have developed first-order distributed algorithms that are capable of computing second-order stationary solutions (which under suitable conditions become local optimal solutions). Other second-order distributed algorithms such as [27,28] are design for convex problems, and they utilize high-order Hessian information about local problems.

## 1.3  Lower and upper rate bounds analysis

Despite all the recent interests and contributions in this field, one major question remains open:

> **(Q)**  What is the *best* convergence rate achievable by *any* distributed algorithms for the non-convex problem (1)?

Question **(Q)** seeks to find a "best convergence rate", which is a characterization of the smallest number of iterations required to achieve certain *high-quality solutions*, among all distributed algorithms. Clearly, understanding **(Q)** provides fundamental insights to distributed optimization and information processing. For example, the answer to **(Q)** offers meaningful optimal estimates on the total amount of communication

and computation effort required to achieve a given level of accuracy. Further, the identified optimal strategies capable of attaining the best convergence rates will also help guide the practical design of distributed information processing algorithms.

Question (**Q**) is easy to state, but formulating it rigorously is quite involved and a number of delicate issues have to be clarified. Below we provide a high level discussion on some of these issues.

**(1) Fix Problem and Network Classes.** A class of problems $\mathcal{P}$ and networks $\mathcal{N}$ of interest should be fixed. Roughly speaking, in this work, we will fix $\mathcal{P}$ to be the family of smooth unconstrained problem (1), and $\mathcal{N}$ to be the set of *connected* and *unweighted* graphs with finite number of nodes.

**(2) Characterize High-Quality Solutions.** For a properly defined error constant $\epsilon > 0$, one needs to define a *high-quality solution* in distributed and non-convex setting. Differently from the centralized case, the following questions have to be addressed: Should the solution quality be evaluated based on the *averaged* iterates among all the agents, or on the individual iterates? Shall we include some *consensus measure* in the solution characterization? Different solution notion could potentially lead to different lower and upper rate bounds.

**(3) Fix Algorithm Classes.** A class of algorithms $\mathcal{A}$ has to be fixed. In the classical complexity analysis in (centralized) optimization, it is common to define the class of algorithms by the information structures that they utilize [29]. In the distributed and non-convex setting, it is necessary to specify *both* the function information that can be used by individual nodes, as well as the communication protocols that are allowed.

**(4) Develop Sharp Upper Bounds.** It is necessary to develop algorithms within class $\mathcal{A}$, which possess provable and sharp global convergence rate for problem/network class $(\mathcal{P}, \mathcal{N})$. These algorithms provide achievable *upper bounds* on the global convergence rates.

**(5) Identify Lower Bounds.** It is important to characterize the *worst* rates achievable by *any* algorithm in class $\mathcal{A}$ for problem/network class $(\mathcal{P}, \mathcal{N})$. This task involves identifying instances in $(\mathcal{P}, \mathcal{N})$ that are difficult for algorithm class $\mathcal{A}$.

**(6) Match Lower and Upper Bounds.** The key task is to investigate whether the developed algorithms are *rate optimal*, in the sense that rate upper bounds derived in **(4)** match the *worst-case* lower bounds developed in **(5)**. Roughly speaking, matching two bounds requires that for the class of problem and networks $(\mathcal{P}, \mathcal{N})$, the following quantities should be matched between the lower and upper bounds: *i)* the order of the error constants $\epsilon$; *ii)* the order of problem parameters such as $M$, or that of network parameters such as the spectral gap, diameter, etc.

Convergence rate analysis (aka iteration complexity analysis) for convex problems dates back to Nesterov, Nemirovsky and Yudin [30, 31], in which lower bounds and optimal *first-order* algorithms have been developed; also see [32]. In recent years, many accelerated *first-order* algorithms achieving those lower bounds for different kinds of convex problems have been derived; see e.g., [33–35], including those developed for distributed convex optimization [36]. In those works, the problem is to optimize $\min_x f(x)$ with convex $f$, the optimality measure used is $f(x) - f(x^*)$, and the lower bound can be expressed as [32, Theorem 2.2.2]

$$f(x^t) - f(x^*) \leq \frac{\|x^0 - x^*\|L}{(t+2)^2}, \tag{3}$$

where $L$ is the Lipschitz constant for $\nabla f$; $x^*$ (resp. $x^0$) is the global optimal solution (resp. the initial solution); $t$ is the iteration index. Therefore to achieve $\epsilon$-optimal solution in which $f(x^t) - f(x^*) \leq \epsilon$, one needs $\sqrt{\frac{\|x^* - x^0\|L}{\epsilon}}$ iterations. Recently the above approach has been extended to distributed strongly convex optimization in [37]. In particular, the authors consider problem (1) in which each $f_i$ is strongly

| Network Instances | Problem Classes | | Rate Achieving Algorithm |
|---|---|---|---|
| | Uniform Lipschitz $U$ | Non-uniform Lipschitz $\{L_i\}$ | |
| Complete/Star | $\mathcal{O}(U/\epsilon)$ | $\mathcal{O}(1/\epsilon \times \sum_i L_i/M)$ | D-GPDA (proposed) |
| Random Geometric | $\widetilde{\mathcal{O}}(U\sqrt{M}/(\sqrt{\log M}\epsilon))$ | $\widetilde{\mathcal{O}}(\sqrt{M}/(\sqrt{\log(M)}\epsilon) \times \sum_i L_i/M)$ | xFILTER (proposed) |
| Path/Circle | $\widetilde{\mathcal{O}}(UM/\epsilon)$ | $\widetilde{\mathcal{O}}(M/\epsilon \times \sum_i L_i/M)$ | xFILTER (proposed) |
| Grid | $\widetilde{\mathcal{O}}(U\sqrt{M}/\epsilon)$ | $\widetilde{\mathcal{O}}(\sqrt{M}/\epsilon \times \sum_i L_i/M)$ | xFILTER (proposed) |
| Centralized | $\mathcal{O}(U/\epsilon)$ | $\mathcal{O}(1/\epsilon \times \sum_i L_i/M)$ | Gradient Descent |

**Table 1:** The main results of the paper when specializing to a few popular graphs. The entries show the best rate bounds achieved by the proposed algorithms (either D-GPDA or xFILTER) for a number of specific graphs and problem class; $L_i$ is the Lipschitz constant for $\nabla f_i$ [see (4)]; for the uniform case $U = L_1, \cdots, L_M$. For the uniform Lipschitz the lower rate bounds derived for the particular graph matches the upper rate bounds (we only show the latter in the table). The last row shows the rate achieved by the centralized gradient descent algorithm. The notation $\tilde{\mathcal{O}}$ denotes big $\mathcal{O}$ with some polynomial in logarithms, i.e, use $\widetilde{\mathcal{O}}$ to denote $\mathcal{O}(\log(M))$ where $M$ is the problem dimension.

convex, and they provide lower and upper rate bounds for a class of algorithms in which the local agents can utilize both $\nabla f_i(x)$ and its Fenchel conjugate $\nabla^* f_i(x)$. We note that this result is not directly related to the class of "first-order" method, since beyond the first-order gradient information, the Fenchel conjugate $\nabla^* f_i(x)$ is also needed, but computing this quantity requires performing certain exact minimization, which itself involves solving a strongly convex optimization problem. Other related works in this direction also include [38] and [39]. In particular, the work [39] is a non-smooth extension of [37], where the lower complexity bound under the Lipschitz continuity of the global and local objective function are discussed and the optimal algorithm is proposed.

When the problem becomes non-convex, the size of the gradient function can be used as a measure of solution quality. In particular, let $h_T^* := \min_{0 \le t \le T} \|\nabla f(x^t)\|^2$, then it has been shown that the classical (centralized) gradient descent (GD) method achieves the following rate [32, page 28]

$$h_T^* \le \frac{c_0 L(f(x^0) - f(x^*))}{T + 1}, \text{ where } c_0 > 0 \text{ is some constant.}$$

It has been shown in [40] that the above rate is (almost) tight for GD. Recently, [41] has further shown that the above rate is optimal for *any* first-order methods that only utilize the gradient information, when applied to problems with Lipschitz gradient. However, no lower bound analysis has been developed for distributed non-convex problem (19); there are even not many algorithms that provide achievable upper rate bounds (except for the recent works [12, 15, 42, 43]), not to mention any analysis on the tightness/sharpness of these upper bounds.

## 1.4 Contribution of this work

In this work, we address various issues that arise in answering (**Q**). Our main contributions are given below:

**1)** We identify a class of non-convex problems and networks $(\mathcal{P}, \mathcal{N})$, a class of distributed first-order algorithms $\mathcal{A}$, and rigorously define the $\epsilon$-optimality gap that measures the progress of the algorithms;

**2)** We develop the first lower complexity bound for class $\mathcal{A}$ to solve class $(\mathcal{P}, \mathcal{N})$: To achieve $\epsilon$-optimality, it is necessary for any $a \in \mathcal{A}$ to perform $\mathcal{O}(1/\sqrt{\xi(\mathcal{G})} \times \bar{L}/\epsilon)$ rounds of communication among all the nodes, where $\xi(\mathcal{G})$ represents certain spectral gap of the graph Laplacian matrix, and $\bar{L}$ is the averaged Lipschitz constants of the gradients of local functions. On the other hand, it is necessary for any such algorithm to

perform $\mathcal{O}(\bar{L}/\epsilon)$ rounds of computation among all the nodes.

**3)** We design two algorithms belonging to $\mathcal{A}$, one based on primal-dual optimization scheme, the other based on a novel *approximate filtering -then- predict and tracking* (xFILTER) strategy, both of which achieve $\epsilon$-optimality condition with provable global rates [in the order of $\mathcal{O}(1/\epsilon)$];

**4)** We show that the xFILTER is an optimal method in $\mathcal{A}$ for problem class $(\mathcal{P},\mathcal{N})$ as well as a number of its refinements, in that they precisely achieve the lower complexity bounds that we derived (up to a ploylog factor).

In Table 1, we specialize some key results developed in the paper to a few popular graphs.

**Notations.** For a given symmetric matrix $B$, we use $\lambda_{\max}(B)$, $\lambda_{\min}(B)$ and $\underline{\lambda}_{\min}(B)$ to denote the maximum, the minimum and the minimum *nonzero* eigenvalues; We use $I_P$ to denote an identity matrix with size $P$, and use $\otimes$ to denote the Kronecker product. We use $[M]$ to denote the set $\{1, \cdots, M\}$. For a vector $x$ we use $x[i]$ to denote its $i$th element. We use $\widetilde{\mathcal{O}}$ to denote $\mathcal{O}(\log(M))$ where $M$ is the problem dimension. We use $i \sim j$ to denote two connected nodes $i$ and $j$, i.e., for a graph $\mathcal{G} := \{\mathcal{V},\mathcal{E}\}$, $i \sim j$ if $i \neq j$, and $(i,j) \in \mathcal{E}$.

# 2  Preliminaries

## 2.1  The class $\mathcal{P}$, $\mathcal{N}$, $\mathcal{A}$

We present the classes of problems, networks and algorithms to be studied, as well as some useful results. We parameterize these classes using a few key parameters so that we can specify their subclasses when needed.

**Problem Class.** A problem is in class $\mathcal{P}_L^M$ if it satisfies the following conditions.

A1. The objective is an average of $M$ functions; see (1).

A2. Each component function $f_i(x)$'s has Lipschitz gradient:

$$\|\nabla f_i(x_i) - \nabla f_i(z_i)\| \le L_i \|x_i - z_i\|, \ \forall \ x_i, z_i \in \mathbb{R}^S, \ \forall \ i, \tag{4}$$

where $L_i \ge 0$ is the *smallest* positive number such that the above inequality holds true. Define $\bar{L} := \frac{1}{M}\sum_{i=1}^M L_i$, $L_{\max} := \max_i L_i$, and $L_{\min}$ similarly.

Define the matrix of Lipschitz constants as:

$$L := \mathrm{diag}([L_1, \cdots, L_M]) \otimes I_S \in \mathbb{R}^{MS \times MS}. \tag{5}$$

A3. The function $f(x)$ is lower bounded over $x \in \mathbb{R}^{MS}$, i.e.,

$$\underline{f} := \inf_x f(x) > -\infty. \tag{6}$$

These assumptions are rather mild. For example an $f_i$ satisfies [A2-A3] is not required to be second-order differentiable. Below we provide a few non-convex functions that satisfy Assumption [A2-A3], and each of those can be the component function $f_i$'s. Note that the first four functions are of particular interest in learning neural networks, as they are commonly used as activation functions.

**(1)** The sigmoid function is given by $\mathrm{sigmoid}(x) = \frac{1}{1+e^{-x}}$. We have $\mathrm{sigmoid}(x) \ge 0$, $\mathrm{sigmoid}''(x) \in (-1,1)$, therefore [A2-A3] are true with $L \le 1$.

**(2)** The arctan function satisfies $\arctan(x) \in (-\frac{\pi}{2}, \frac{\pi}{2})$, $\arctan''(x) = \frac{-2x}{(x^2+1)^2} \in [-1, 1]$. So [A2-A3] hold with $L \leq 1$.

**(3)** The tanh function satisfies $\tanh(x) \geq -1$, $\tanh''(x) \in [-1, 1]$, so [A2-A3] hold with $L \leq 1$.

**(4)** The logit function is related to the tanh function as follows

$$2\mathrm{logit}(x) = \frac{2e^x}{e^x + 1} = 1 + \tanh(x/2),$$

then Assumptions [A2-A3] are again satisfied.

**(5)** The $\log(1 + x^2)$ function has applications in structured matrix factorization [44]. Clearly it is lower bounded. Its second-order derivative is also bounded.

**(6)** Other functions like $\sin(x)$, $\mathrm{sinc}(x)$, $\cos(x)$ are easy to verify. Consider $f(x) := -x_1 x_2 + (x_1 - 1)_+^2 + (-x_1 - 1)_+^2$ where $(z)_+^2 := \max\{0, z\}^2$. This function is interesting because it is not second-order differentiable; nonetheless we can verify that [A2-A3] are satisfied with $L = \sqrt{2} + 1$.

**Network Class.** Let $\mathcal{N}$ denote a class of networks represented by an *undirected* and *unweighted* graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, with $|\mathcal{V}| = M$ vertices and $|\mathcal{E}| = E$ edges, and edge weights all being 1. In this paper the term 'network' and 'graph' will be used interchangeably. Also, we use $\mathcal{N}_D^M$ to denote a class of network similarly as above, but with $M$ nodes and a diameter of $D$, defined below [where dist$(\cdot)$ indicates the distance between two nodes]:

$$D := \max_{u,v \in \mathcal{V}} \mathrm{dist}(u, v). \tag{7}$$

Following the convention in [45], we define a number of graph related quantities below. First, define the *degree* of node $i$ as $d_i$, and define the averaged degree as:

$$\bar{d} := \frac{1}{M} \sum_{i=1}^{M} d_i. \tag{8}$$

Define the incidence matrix (IM) $A \in \mathbb{R}^{E \times M}$ as follows: if $e \in \mathcal{E}$ and it connects vertex $i$ and $j$ with $i > j$, then $A_{ev} = 1/\sqrt{d_v}$ if $v = i$, $A_{ev} = -1/\sqrt{d_v}$ if $v = j$ and $A_{ev} = 0$ otherwise; see the definition in [45, Theorem 8.3]. Using these definitions, the *graph Laplacian matrix* and the *degree matrix* are defined as follows (see [45, Section 1.2]):

$$\mathcal{L} := A^T A \in \mathbb{R}^{M \times M}, \quad \text{and} \quad P := \mathrm{diag}[d_1, \cdots, d_M] \in \mathbb{R}^{M \times M}. \tag{9}$$

In particular, the elements of the Laplacian are given as:

$$[\mathcal{L}]_{ij} = \begin{cases} 1 & \text{if } i = j \\ -\frac{1}{\sqrt{d_i d_j}} & \text{if } i \sim j, i \neq j \\ 0 & \text{otherwise.} \end{cases}$$

We note that the graph Laplacian defined here is sometimes known as the *normalized* graph Laplacian in the literature, but throughout this paper we follow the convention used in the classical work [45] and simply refer it as the *graph Laplacian*. For convenience, we also define a scaled version of the IM:

$$F := AP^{1/2} \in \mathbb{R}^{E \times M}. \tag{10}$$

6

It is known that IM and scaled IM satisfy the following (where $\mathbb{1} \in \mathbb{R}^M$ is an all one vector):

$$F\mathbb{1} = AP^{1/2}\mathbb{1} = 0. \tag{11}$$

Define the second smallest eigenvalue of $\mathcal{L}$, as $\underline{\lambda}_{\min}(\mathcal{L})$:

$$\underline{\lambda}_{\min}(\mathcal{L}) = \inf_{x:\sum_{i=1}^M x_i d_i = 0, x \neq 0} \frac{x^T \mathcal{L} x}{\sum_{i=1}^M x_i^2 d_i}. \tag{12}$$

Then the *spectral gap* of the graph $\mathcal{G}$ can be defined below:

$$\xi(\mathcal{G}) = \frac{\lambda_{\min}(\mathcal{L})}{\lambda_{\max}(\mathcal{L})} \leq 1. \tag{13}$$

**Algorithm Class.** Define the *neighbor set* for node $i \in \mathcal{E}$ as

$$\mathcal{N}_i := \{i \mid i \sim j, j \neq i\}. \tag{14}$$

We say that a *distributed, first-order* algorithm is in class $\mathcal{A}$ if it satisfies the following conditions.

B1. At iteration 0, each node can obtain some network related constants, such as $M$, $D$, eigenvalues of the graph Laplacian $\mathcal{L}$, etc.

B2. At iteration $t + 1$, *each node $i \in [M]$* first conducts a communication step by broadcasting the local $x_i^t$ to all its neighbors, through a function $Q_i^t(\cdot) : \mathbb{R}^S \to \mathbb{R}^S$. Then each node will generate the new iterate, by combining the received message with its past gradients using a function $W_i^t(\cdot)$:

$$v_i^t = \underbrace{Q_i^t(x_i^t)}_{\text{communication step}}, \; x_i^{t+1} \in \underbrace{W_i^t\left(\{\{v_j^k\}_{j\in\mathcal{N}_i}, \nabla f_i(x_i^k), x_i^k\}_{k=1}^t\right)}_{\text{computation step}}. \tag{15}$$

In this work, we will focus on the case where the $Q_i^t(\cdot)$'s and $W_i^t(\cdot)$'s are linear operators.

Clearly $\mathcal{A}$ belongs to the class of *first-order* methods because only local gradient information is used. It is also a class of *distributed* algorithms because at each iteration the nodes only communicate with their immediate neighbors.

Additionally, in practical distributed algorithms such as DSG, ADMM or EXTRA, nodes are dictated to use a *fixed strategy* to linearly combine all its neighbors' information. To model such a requirement, below we consider a slightly restricted algorithm class $\mathcal{A}'$, where we require each node to use the same coefficients to combine its neighbors (note that allowing the nodes to use a fixed but *arbitrary* linear combination is also possible, but the resulting analysis will be more involved).

In particular, we say that a *distributed, first-order* algorithm is in $\mathcal{A}'$ if it satisfies B1 and the following:

B2'. At iteration $t + 1$, *each node $i \in [M]$* performs:

$$v_i^t = Q_i^t(x_i^t), \; x_i^{t+1} \in W_i^t\left(\{\sum_{j\in\mathcal{N}_i} v_j^t, \nabla f_i(x_i^k), x_i^k\}_{k=1}^t\right). \tag{16}$$

We remark that, in both algorithm classes, *one round* of communication occurs at each iteration, where *each* node broadcasts its local variable $x_i^t$ once. Therefore, the total iteration number is the same as the total communication rounds. However, the total times that the entire gradient $\{\nabla f_i(x_i)\}_{i=1}^M$ is evaluated could be *smaller* than the total iteration number/communication rounds. This is because when we compute $x_i^{t+1}$, the operation $W_i^t(\cdot)$ can set the coefficient in front of $\nabla f_i(x_i^\tau)$ to be zero, effectively *skipping* the local gradient computation.

## 2.2 Solution Quality Measure

Next we provide definitions for the quality of the solution. Note that since we consider using first-order methods to solve non-convex problems, it is expected that in the end some first-order stationary solution with small $\|\nabla f\|$ will be computed.

Our first definition is related to a *global variable* $y^t \in \mathbb{R}^S$. We say that $y^t$ is a *global $\epsilon$-solution* if the following holds:

$$y^t \in \text{span}\left\{x_i^t\right\}_{i=1}^M, \quad \min_{t \in [T]} \|\nabla g(y^t)\|^2 \leq \epsilon. \tag{17}$$

This definition is conceptually simple and it is identical to the centralized criteria in Section 1.3. However it has the following issues. First, no global variable $y^t$ will be formed in the entire network, so criteria (17) is difficult to evaluate. Second, there is no characterization of how close the local variables $x_i^t$'s are. To see the second point, consider the following toy example.

*Example 1:* Consider a network with $M = 2$ and $f_1(y) = -y^2$ and $f_2(y) = y^2$. Suppose that the local variables take the following values: $x_1^T = -10$ and $x_2^T = 10$. Then if we pick $y^T = (x_1^T + x_2^T)/2 = 0$, we have

$$\nabla g(y^T) = \frac{1}{2}(\nabla f_1(y^T) + \nabla f_2(y^T)) = 0.$$

This suggests that at iteration $T$, there exists one linear combination that makes measure (17) precisely zero. However one can hardly say that the current solution $(x_1^T, x_2^T) = (-10, 10)$ is a good solution for problem (2). □

To address the above issue, we provide a second definition which is directly related to *local variables* $\{x_i \in \mathbb{R}^S\}_{i=1}^M$. At a given iteration $T$, we say that $\{x_i^T\}$ is a *local $\epsilon$-solution* if the following holds:

$$h_T^* := \min_{t \in [T]} \left\| \sum_{i=1}^M \frac{\nabla f_i(x_i^t)}{M} \right\|^2 + \frac{1}{M\lambda_{\min}(P^{1/2}\mathcal{L}P^{1/2})} \sum_{(i,j):i\sim j} \sqrt{L_i L_j}\|x_i^t - x_j^t\|^2 \leq \epsilon. \tag{18}$$

Clearly this definition takes into consideration the consensus error as well as the size of the local gradients. When applied to Example 1, this measure will be large. Note that the constant $\frac{1}{M\lambda_{\min}(P^{1/2}\mathcal{L}P^{1/2})}$ is needed to balance the two different measures. Also note that the "$\min_{t \in [T]}$" operation is needed to track the best solution obtained before iteration $T$, because the quantity inside this operation may not be monotonically decreasing.

In our work we will focus on providing answers to the following specific version of question (**Q**):

For *any* given $\epsilon > 0$, what is the minimum iteration $T$ (as a function of $\epsilon$) needed for any algorithm in class $\mathcal{A}$ (or class $\mathcal{A}'$) to solve instances in classes $(\mathcal{P}, \mathcal{N})$, so to achieve $h_T^* \leq \epsilon$?

## 2.3 Some Useful Facts and Definitions

Below we provide a few facts about the above classes.

**On Lipschitz constants.** Assume that each $f_i$ has Lipschitz continuous gradient with constant $L_i$ in (4). Then we have :

$$\|\nabla \bar{f}(y_1) - \nabla \bar{f}(y_2)\| \leq \sum_{i=1}^{M} \frac{1}{M} L_i \|y_1 - y_2\| := \bar{L}\|y_1 - y_2\|, \ \forall \ y_1, \ y_2 \in \mathbb{R}^S, \tag{19}$$

where $\bar{L}$ is the average of the local Lipschitz gradients. We also have the following

$$\|\nabla f(x) - \nabla f(z)\|^2 = \frac{1}{M^2} \sum_{i=1}^{M} \|\nabla f_i(x_i) - \nabla f_i(z_i)\|^2, \ \forall \ x_i, \ z_i \in \mathbb{R}^S$$

which implies

$$\|\nabla f(x) - \nabla f(z)\| \leq \frac{1}{M} \|L(x - z)\|, \quad \forall \ x, z \in \mathbb{R}^{MS}, \tag{20}$$

where the matrix $L$ is defined in (5).

**On Quantities for Graph $\mathcal{G}$.** This section presents a number of properties for a given graph $\mathcal{G}$. Define the following matrices:

$$\Sigma := \mathrm{diag}[\sigma_1, \cdots, \sigma_E] \succ 0, \ \Upsilon := \mathrm{diag}([\beta_1, \cdots, \beta_M]) \succ 0. \tag{21}$$

Define $B \in \mathbb{R}^{E \times M} = |F|$ where the absolute value is taken component-wise. Then we have the following:

$$\frac{1}{2}\left(F^T F + B^T B\right) = P = \mathrm{diag}[d_1, \cdots, d_M] \in \mathbb{R}^{M \times M} \tag{22}$$

$$\frac{1}{2}\left(F^T \Sigma^2 F + B^T \Sigma^2 B\right) = \mathrm{diag}\left(\left\{\sum_{j:i\sim j} \sigma_{ij}^2\right\}_{j\in\mathcal{N}}\right) := \Delta,$$

where $P$ is the *degree matrix* defined in (9).

For two diagonal matrices $\Upsilon^2$ and $\Sigma^2$ of appropriate sizes, the *generalized Laplacian* (GL) matrix is defined as:

$$\mathcal{L}_G = \Upsilon^{-1} F^T \Sigma^2 F \Upsilon^{-1}, \tag{23}$$

and its elements are given by:

$$[\mathcal{L}_G]_{ij} = \begin{cases} \frac{\sum_{q:i\sim q} \sigma_{iq}^2}{\beta_i^2} & \text{if } i = j \\ -\frac{\sigma_{ij}^2}{\beta_i \times \beta_j} & \text{if } (ij) \in \mathcal{E}, i \neq j \\ 0 & \text{otherwise} \end{cases}.$$

Define a diagonal matrix $K \in \mathbb{R}^{E \times E}$ as below:

$$[K]_{e,q} = \begin{cases} \sqrt{L_i L_j} & \text{if } e = q, \text{ and } e = (i,j) \\ 0 & \text{otherwise} \end{cases}. \tag{24}$$

Then when specializing $\Upsilon = P^{1/2}L^{1/2}$ and $\Sigma^2 = K$, the GL matrix becomes:

$$\widetilde{\mathcal{L}} := L^{-1/2}P^{-1/2}F^T KFP^{-1/2}L^{-1/2}. \tag{25}$$

Note that if any diagonal element in the matrix $L$ is zero, then $L^{-1}$ denotes the Moore - Penrose matrix pseudoinverse. Similarly, when specializing $\Upsilon = L^{1/2}$ and $\Sigma^2 = K$, then the GL matrix becomes:

$$\widehat{\mathcal{L}} := L^{-1/2}F^T KFL^{-1/2}. \tag{26}$$

These matrices will be used later in our derivations.

Below we list some useful results about the Laplacian matrix [45–47]. First, all eigenvalues of $\mathcal{L}$ lie in the interval $[0, 2]$. Also because $\underline{\lambda}_{\min}(\mathcal{L}) = \underline{\lambda}_{\min}(P^{-1/2}F^T FP^{-1/2})$, we have

$$\underline{\lambda}_{\min}(\mathcal{L}) \leq \underline{\lambda}_{\min}(F^T F). \tag{27}$$

Also we have that [45, Lemma 1.9]

$$\underline{\lambda}_{\min}(\mathcal{L}) \geq \frac{1}{D \sum_i d_i}. \tag{28}$$

The eigenvalues of $\mathcal{L}$ for a number of special graphs are given below:

1) **Complete Graph:** The eigenvalues are 0 and $M/(M-1)$ (with multiplicity $M-1$), so $\xi(\mathcal{G}) = 1$;

2) **Star Graph:** The eigenvalues are 0 and 1 (with multiplicity $M-2$), and 2, so $\xi(\mathcal{G}) = 1/2$;

3) **Path Graph:** The eigenvalues are $1 - \cos(\pi m/(M-1))$ for $m = 0, 1, \cdots, M-1$, and $\xi(\mathcal{G}) \geq 1/M^2$.

4) **Cycle Graph:** The eigenvalues are $1 - \cos(2\pi m/M)$ for $m = 0, 1, \cdots, M-1$, and $\xi(\mathcal{G}) \geq 1/M^2$.

5) **Grid Graph:** The grid graph is obtained by placing the nodes on a $\sqrt{M} \times \sqrt{M}$ grid, and connecting nodes to their nearest neighbors. We have $\xi(\mathcal{G}) \geq 1/M$.

6) **Random Geometric Graph:** Place the nodes uniformly in $[0, 1]^2$ and connect any two nodes separated by a distance less than a radius $R \in (0, 1)$. Then if the connectivity radius $R$ satisfies [47]

$$R = \Omega\left(\sqrt{\log^{1+\epsilon}(M)/M}\right), \quad \text{for any } \epsilon > 0, \tag{29}$$

then with high probability

$$\xi(\mathcal{G}) = \mathcal{O}\left(\frac{\log(M)}{M}\right). \tag{30}$$

## 3  Lower Complexity Bounds

In this section we develop the lower complexity bounds for algorithms in class $\mathcal{A}$ to solve problems $\mathcal{P}_L^M$ over network $\mathcal{N}$. We will mainly focus on the case where $f_i$'s have uniform Lipschitz constants, that is, we

**Figure 1:** The path graph used in our construction.

assume that

$$L_i = U, \quad \forall\, i \in [M],$$

and we denote the resulting problem class as $\mathcal{P}_U^M$. At the end of this section, generalization to the non-uniform case will be briefly discussed.

Our proof combines ideas from the classical proof in Nesterov [29], as well as two recent constructions [41] (for centralized non-convex problems) and [37] (for strongly convex distributed problems). Our construction differs from the previous works in a number of ways, in particular, the constructed functions are only first-order differentiable, but not second-order differentiable. Further, we use the local-$\epsilon$ solution (18) to measure the quality of the solution, which makes the analysis more involved compared with the existing *global* error measures in [29, 37, 41].

To begin with, we construct the following two non-convex functions

$$h(x) := \frac{1}{M} \sum_{i=1}^{M} h_i(x_i), \quad f(x) := \frac{1}{M} \sum_{i=1}^{M} f_i(x_i), \tag{31}$$

as well as the corresponding versions that evaluate on a "centralized" variable $y$

$$\bar{h}(y) := \frac{1}{M} \sum_{i=1}^{M} h_i(y), \quad \bar{f}(y) := \frac{1}{M} \sum_{i=1}^{M} f_i(y). \tag{32}$$

Here we have $x_i \in \mathbb{R}^T$, for all $i$, $y \in \mathbb{R}^T$, and $x := (x_1, \cdots x_M) \in \mathbb{R}^{TM \times 1}$. Later we make our construction so that functions $h$ and $\bar{h}$ are easy to analyze, while $f$ and $\bar{f}$ will be in the desired function class in $\mathcal{P}_U^M$. Without loss of generality, in the construction we will assume $\nabla f_i$ will be Lipschitz with constant $U \in (0, 1)$, for all $i \in [M]$.

## 3.1 Path Graph ($D = M - 1$)

First we consider the extreme case in which the nodes form a path graph with $M$ nodes and each node $i$ has its own local function $h_i$, shown in Figure 1. For notational simplicity assume that $M$ is a multiple of 3, that is $M = 3C$ for some integer $C > 0$. Also assume that $T$ is an odd number without loss of generality.

Let us define the component functions $h_i$'s in (31) as follows.

$$h_i(x_i) = \begin{cases} \Theta(x_i, 1) + 3 \displaystyle\sum_{j=1}^{\lfloor T/2 \rfloor} \Theta(x_i, 2j), & i \in \left[1, \dfrac{M}{3}\right] \\[4mm] \Theta(x_i, 1), & i \in \left[\dfrac{M}{3} + 1, \dfrac{2M}{3}\right] \\[4mm] \Theta(x_i, 1) + 3 \displaystyle\sum_{j=1}^{\lfloor T/2 \rfloor} \Theta(x_i, 2j+1), & i \in \left[\dfrac{2M}{3} + 1, M\right] \end{cases} \tag{33}$$

11

**Figure 2:** The functional value, and derivatives of $\Psi$.



**Figure 3:** The functional value, and derivatives of $\Phi$.

where we have defined the following functions

$$\Theta(x_i, j) := \Psi(-x_i[j-1])\Phi(-x_i[j]) - \Psi(x_i[j-1])\Phi(x_i[j]), \ \forall \ j \geq 2 \tag{34a}$$

$$\Theta(x_i, 1) := -\Psi(1)\Phi(x_i[1]). \tag{34b}$$

The component functions $\Psi, \Phi : \mathbb{R} \to \mathbb{R}$ are given as below

$$\Psi(w) := \begin{cases} 0 & w \leq 0 \\ 1 - e^{-w^2} & w > 0, \end{cases} \quad \text{and} \quad \Phi(w) := 4\arctan w + 2\pi.$$

Suppose $x_1 = x_2 = \cdots = x_M = y$, then the average function becomes:

$$\bar{h}(y) := \frac{1}{M} \sum_{j=1}^{M} h_i(y) = \Theta(y, 1) + \sum_{i=2}^{T} \Theta(y, i)$$

$$= -\Psi(1)\Phi(y[1]) + \sum_{i=2}^{T} \left[ \Psi(-y[i-1]) \Phi(-y[i]) - \Psi(y[i-1]) \Phi(y[i]) \right].$$

Further for a given error constant $\epsilon > 0$ and a given averaged Lipschitz constant $U \in (0,1)$, let us define

$$f_i(x_i) := \frac{150\pi\epsilon}{U} h_i \left( \frac{x_i U}{75\pi\sqrt{2\epsilon}} \right). \tag{35}$$

Therefore we also have, if $x_1 = x_2 = \cdots = x_M = y$, then

$$\bar{f}(y) := \frac{1}{M} \sum_{i=1}^{M} f_i(y) = \frac{150\pi\epsilon}{U} \bar{h} \left( \frac{yU}{75\pi\sqrt{2\epsilon}} \right). \tag{36}$$

First we present some properties of the component functions $h_i$'s.

12

**Figure 4:** The functional value for $\Theta(w, v) = \Psi(w)\Phi(v)$.

**Lemma 3.1** *The functions $\Psi$ and $\Phi$ satisfy the following.*

1. *For all $w \leq 0$, $\Psi(w) = 0$, $\Psi'(w) = 0$.*

2. *The following bounds hold for the functions and their first and second-order derivatives:*

$$0 \leq \Psi(w) < 1, \quad 0 \leq \Psi'(w) \leq \sqrt{\frac{2}{e}}, \quad -\frac{4}{e^{\frac{3}{2}}} \leq \Psi''(w) \leq 2, \quad \forall w > 0$$

$$0 < \Phi(w) < 4\pi, \quad 0 < \Phi'(w) \leq 4, \quad -\frac{3\sqrt{3}}{2} \leq \Phi''(w) \leq \frac{3\sqrt{3}}{2}, \quad \forall w \in \mathbb{R}$$

3. *The following key property holds:*

$$\Psi(w)\Phi'(v) > 1, \quad \forall \, w \geq 1, \, |v| < 1. \tag{37}$$

4. *The function $h$ is lower bounded as follows:*

$$h_i(0) - \inf_{x_i} h_i(x_i) \leq 10\pi T, \; h(0) - \inf_x h(x) \leq 10\pi T.$$

5. *The first-order derivative of $\bar{h}$ (resp. $h_j$) is Lipschitz continuous with constant $\ell = 75\pi$ (resp. $\ell_j = 75\pi$, $\forall \, i$).*

**Proof.** Property 1) is obviously true.

To prove Property 2), note that following holds for $w > 0$:

$$\Psi(w) = 1 - e^{-w^2}, \quad \Psi'(w) = 2e^{-w^2}w, \quad \Psi''(w) = 2e^{-w^2} - 4e^{-w^2}w^2, \; \forall \, w > 0. \tag{38}$$

Obviously, $\Psi(w)$ is an increasing function over $w > 0$, therefore the lower and upper bounds are $\Psi(0) = 0, \Psi(\infty) = 1$; $\Psi'(w)$ is increasing on $[0, \frac{1}{\sqrt{2}}]$ and decreasing on $[\frac{1}{\sqrt{2}}, \infty]$, where $\Psi''(\frac{1}{\sqrt{2}}) = 0$, therefore the lower and upper bounds are $\Psi'(0) = \Psi'(\infty) = 0, \Psi'(\frac{1}{\sqrt{2}}) = \sqrt{\frac{2}{e}}$; $\Psi''(w)$ is decreasing on $(0, \sqrt{\frac{3}{2}}]$ and increasing on $[\sqrt{\frac{3}{2}}, \infty)$ [this can be verified by checking the signs of $\Psi'''(w) = 4e^{-w^2}w(2w^2 - 3)$ in these intervals]. Therefore the lower and upper bounds are $\Psi''(\sqrt{\frac{3}{2}}) = -\frac{4}{e^{\frac{3}{2}}}, \Psi''(0^+) = 2$, i.e.,

$$0 \leq \Psi(w) < 1, \quad 0 \leq \Psi'(w) \leq \sqrt{\frac{2}{e}}, \quad -\frac{4}{e^{\frac{3}{2}}} \leq \Psi''(w) \leq 2, \quad \forall w > 0.$$

Further, for all $w \in \mathbb{R}$, the following holds:

$$\Phi(w) = 4\arctan w + 2\pi, \quad \Phi'(w) = \frac{4}{w^2 + 1}, \quad \Phi''(w) = -\frac{8w}{(w^2 + 1)^2}. \tag{39}$$

Similarly, as above, we can obtain the following bounds:

$$0 < \Phi(w) < 4\pi, \quad 0 < \Phi'(w) \leq 4, \quad -\frac{3\sqrt{3}}{2} \leq \Phi''(w) \leq \frac{3\sqrt{3}}{2}, \quad \forall w \in \mathbb{R}.$$

We refer the readers to Fig. 2 – Fig. 3 for illustrations of these functions.

To show Property 3), note that for all $w \geq 1$ and $|v| < 1$,

$$\Psi(w)\Phi'(v) > \Psi(1)\Phi'(1) = 2(1 - e^{-1}) > 1$$

where the first inequality is true because $\Psi(w)$ is strictly increasing and $\Phi'(v)$ is strictly decreasing for all $w > 0$ and $v > 0$, and that $\Phi'(v) = \Phi'(|v|)$.

Next we show Property 4). Note that $0 \leq \Psi(w) < 1$ and $0 < \Phi(w) < 4\pi$. Therefore we have $h(0) = -\Psi(1)\Phi(0) < 0$ and using the construction in (33)

$$\inf_{x_i} h_i(x_i) \geq -\Psi(1)\Phi(x_i[1]) - 3 \sum_{j=1}^{\lfloor T/2 \rfloor} \sup_{w,v} \Psi(w)\Phi(v) \tag{40}$$

$$\geq -4\pi - 6\pi T \geq -10\pi T, \tag{41}$$

where the first inequality follows $\Psi(w)\Phi(v) > 0$ and second follows $\Psi(w)\Phi(v) < 4\pi$, we reach the conclusion.

Finally we show Property 5), using the fact that a function is Lipschitz if it is piecewise smooth with bounded derivative. From construction (33), the first-order partial derivative of $h_q(y)$ can be expressed below.

**Case I)** If $i$ is even, we have

$$\frac{\partial h_q}{\partial y[i]} = \begin{cases} 3\left(-\Psi\left(-y[i-1]\right)\Phi'\left(-y[i]\right) - \Psi\left(y[i-1]\right)\Phi'\left(y[i]\right)\right), & q \in [1, \frac{M}{3}] \\ 0, & q \in [\frac{M}{3} + 1, \frac{2M}{3}] \\ 3\left(-\Psi'\left(-y[i]\right)\Phi\left(-y[i+1]\right) - \Psi'\left(y[i]\right)\Phi\left(y[i+1]\right)\right), & q \in [\frac{2M}{3} + 1, M] \end{cases} \tag{42}$$

14

**Case II)** If $i$ is odd but not 1, we have

$$\frac{\partial h_q}{\partial y[i]} = \begin{cases} 3\left(-\Psi'\left(-y[i]\right)\Phi\left(-y[i+1]\right) - \Psi'\left(y[i]\right)\Phi\left(y[i+1]\right)\right), & q \in [1, \frac{M}{3}] \\ 0, & q \in [\frac{M}{3}+1, \frac{2M}{3}] \\ 3\left(-\Psi\left(-y[i-1]\right)\Phi'\left(-y[i]\right) - \Psi\left(y[i-1]\right)\Phi'\left(y[i]\right)\right), & q \in [\frac{2M}{3}+1, M] \end{cases} \quad . \tag{43}$$

**Case III)** If $i = 1$, we have

$$\frac{\partial h_q}{\partial y[1]} = \begin{cases} -\Psi(1)\Phi'(y[1]) + 3\left(-\Psi'\left(-y[1]\right)\Phi\left(-y[2]\right) - \Psi'\left(y[1]\right)\Phi\left(y[2]\right)\right), & q \in [1, \frac{M}{3}] \\ -\Psi(1)\Phi'(y[1]), & q \in [\frac{M}{3}+1, M] \end{cases} \quad . \tag{44}$$

Obviously, $\frac{\partial h_q}{\partial y[i]}$ is a piecewise smooth function for any $i, q$, and it either equals zero or is separated at the non-differentiable point $y[i] = 0$ because of the function $\Psi$.

Further, fix a point $y \in \mathbb{R}^T$ and a unit vector $v \in \mathbb{R}^T$ where $\sum_{i=1}^T v[i]^2 = 1$. Define

$$g_q(\theta; y, v) := h_q(y + \theta v)$$

to be the directional projection of $h_q$ on to the direction $v$ at point $y$. We will show that there exists $\ell > 0$ such that $|g_q''(0; y, v)| \le \ell$ for all $y \ne 0$ (where the second-order derivative is taken with respect to $\theta$).

First we can compute $g_q''(0; y, v)$ as follows:

$$g_q''(0; y, v) = \sum_{i_1, i_2 = 1}^T \frac{\partial^2}{\partial y[i_1] \partial y[i_2]} h_q(y) v[i_1] v[i_2] = \sum_{\delta \in \{0, 1, -1\}} \sum_{i=1}^T \frac{\partial^2}{\partial y[i] \partial y[i+\delta]} h_q(y) v[i] v[i+\delta],$$

where we take $v[0] := 0$ and $v[T+1] := 0$.

The second-order partial derivative of $h_q(y)$ ($\forall y \ne 0$) is given as follows when $i$ is even:

$$\frac{\partial^2 h_q}{\partial y[i] \partial y[i]} = \begin{cases} 3\left(\Psi\left(-y[i-1]\right)\Phi''\left(-y[i]\right) - \Psi\left(y[i-1]\right)\Phi''\left(y[i]\right)\right), & q \in [1, \frac{M}{3}] \\ 0, & q \in [\frac{M}{3}+1, \frac{2M}{3}] \\ 3\left(\Psi''\left(-y[i]\right)\Phi\left(-y[i+1]\right) - \Psi''\left(y[i]\right)\Phi\left(y[i+1]\right)\right), & q \in [\frac{2M}{3}+1, M] \end{cases} \tag{45}$$

$$\frac{\partial^2 h_q}{\partial y[i] \partial y[i+1]} = \begin{cases} 0, & q \in [1, \frac{2M}{3}] \\ 3\left(\Psi'\left(-y[i]\right)\Phi'\left(-y[i+1]\right) - \Psi'\left(y[i]\right)\Phi'\left(y[i+1]\right)\right), & q \in [\frac{2M}{3}+1, M] \end{cases} \tag{46}$$

$$\frac{\partial^2 h_q}{\partial y[i] \partial y[i-1]} = \begin{cases} 3\left(\Psi'\left(-y[i-1]\right)\Phi'\left(-y[i]\right) - \Psi'\left(y[i-1]\right)\Phi'\left(y[i]\right)\right), & q \in [1, \frac{M}{3}] \\ 0, & q \in [\frac{M}{3}+1, M] \end{cases} \quad . \tag{47}$$

By applying Lemma 3.1 – i) [i.e., $\Psi(w) = \Psi'(w) = \Psi''(w) = 0$ for $\forall w \le 0$], we immediately obtain that at least one of the terms $\Psi\left(-y[i-1]\right)\Phi''\left(-y[i]\right)$ or $-\Psi\left(y[i-1]\right)\Phi''\left(y[i]\right)$ is zero. It follows that

$$\Psi\left(-y[i-1]\right)\Phi''\left(-y[i]\right) - \Psi\left(y[i-1]\right)\Phi''\left(y[i]\right) \le \sup_w |\Psi(w)| \sup_v |\Phi''(v)|.$$

Similarly,

$$\Psi''\left(-y[i]\right)\Phi\left(-y[i+1]\right) - \Psi''\left(y[i]\right)\Phi\left(y[i+1]\right) \le \sup_w |\Psi''(w)| \sup_v |\Phi(v)|$$

$$\Psi'\left(-y[i]\right)\Phi'\left(-y[i+1]\right) - \Psi'\left(y[i]\right)\Phi'\left(y[i+1]\right) \le \sup_w |\Psi'(w)| \sup_v |\Phi'(v)|.$$

Therefore, take the maximum over equations (45) to (47) and plug in the above inequalities, we obtain

$$\left|\frac{\partial^2 h_q}{\partial y[i_1]\partial y[i_2]}\right| \le 3 \max\{\sup_w |\Psi''(w)| \sup_v |\Phi(v)|, \sup_w |\Psi(w)| \sup_v |\Phi''(v)|, \sup_w |\Psi'(w)| \sup_v |\Phi'(v)|\}$$

$$= 3\max\left\{8\pi, \frac{3\sqrt{3}}{2}, 4\sqrt{\frac{2}{e}}\right\} < 25\pi, \quad \forall\, i_1 \text{ being even, } \forall\, i_2$$

where the equality comes from Lemma 3.1 – ii).

We can also verify that the above bound for $i$ being odd but not 1 is exactly the same.

When $i = 1$ we have following:

$$\frac{\partial^2 h_q}{\partial y[1]\partial y[1]} = \begin{cases} -\Psi(1)\Phi''(y[1]) + 3\left(-\Psi''\left(-y[1]\right)\Phi\left(-y[2]\right) - \Psi''\left(y[1]\right)\Phi\left(y[2]\right)\right), & q \in [1, \frac{M}{3}] \\ -\Psi(1)\Phi''(y[1]), & q \in [\frac{M}{3}+1, M] \end{cases}$$

$$\frac{\partial^2 h_q}{\partial y[1]\partial y[2]} = \begin{cases} 3\left(-\Psi'\left(-y[1]\right)\Phi'\left(-y[2]\right) - \Psi'\left(y[1]\right)\Phi'\left(y[2]\right)\right), & q \in [1, \frac{M}{3}] \\ 0, & q \in [\frac{M}{3}+1, M] \end{cases}$$

Again by applying Lemma 3.1 – i) and ii),

$$\left|\frac{\partial^2 h_q}{\partial y[1]\partial y[i_2]}\right| \le \max\{\sup_w |\Psi(1)\Phi''(w)| + 3\sup_w |\Psi''(w)| \sup_v |\Phi(v)|, 3\sup_w |\Psi'(w)| \sup_v |\Phi'(v)|\}$$

$$= \max\left\{\frac{3\sqrt{3}}{2}(1 - e^{-1}) + 24\pi, 12\sqrt{\frac{2}{e}}\right\} < 25\pi, \forall\, i_2.$$

Summarizing the above results, we obtain:

$$|g_q''\left(0; y, v\right)| = |\sum_{\delta \in \{0,1,-1\}} \sum_{i=1}^{T} \frac{\partial^2}{\partial y[i]\partial y[i+\delta]} h_q\left(y\right) v[i]v[i+\delta]|$$

$$\le 25\pi \sum_{\delta \in \{0,1,-1\}} |\sum_{i=1}^{T} v[i]v[i+\delta]|$$

$$= 25\pi\left(|\sum_{i=1}^{T} v[i]^2| + 2|\sum_{i=1}^{T} v[i]v[i+1]|\right)$$

$$\le 75\pi \sum_{i=1}^{T} |v[i]^2| = 75\pi.$$

Overall, the first-order derivatives of $h_q$ are Lipschitz continuous for any $q$ with constant $\ell = 75\pi$.

To show the same result for the function $\bar{h}$, we can apply (19). This completes the proof. **Q.E.D.**

The following lemma is a simple extension of the previous result.

**Lemma 3.2** *We have the following properties for the functions $f$ and $\bar{f}$ defined in (36) and (35).*

1. We have $\forall\, x \in \mathbb{R}^{TM \times 1}$

$$f(0) - \inf_x f(x) + \frac{1}{MU}\|d_0\|^2 \le \frac{1650\pi^2\epsilon}{U}T,$$

where we have defined

$$d_0 := [\nabla f_1(0), \cdots, \nabla f_M(0)]. \tag{48}$$

2. We have

$$\|\nabla \bar{f}(y)\| = \sqrt{2\epsilon}\left\|\nabla \bar{h}\left(\frac{yU}{75\pi\sqrt{2\epsilon}}\right)\right\|, \ \forall\, y \in \mathbb{R}^{T \times 1}. \tag{49}$$

3. The first-order derivatives of $\bar{f}$ and that for each $f_j, j \in [M]$ are Lipschitz continuous, with the same constant $U > 0$.

**Proof.** To show that property 1) is true, note that from the definition of $f_i(x_i)$ we have

$$\nabla f_i(x_i) = \sqrt{2\epsilon} \times \nabla h_i\left(\frac{x_i U}{75\pi\sqrt{2\epsilon}}\right).$$

Therefore the following holds:

$$\begin{aligned}
\frac{1}{M}\|d_0\|^2 &= \frac{2\epsilon}{M}\sum_{i=1}^{M}\|\nabla h_i(0)\|^2 \\
&= \frac{2\epsilon}{M}\sum_{i=1}^{M}|\Psi(1)\Phi'(0)|^2 = 32\epsilon(1 - \exp(-1))^2.
\end{aligned} \tag{50}$$

Therefore we have the following:

$$f(0) - \inf_x f(x) + \frac{\|d_0\|^2}{MU} = \frac{150\pi\epsilon}{U}\left(h(0) - \inf_x h(x) + \frac{16(1 - \exp(-1))^2}{75\pi}\right).$$

Then by applying Lemma 3.1 we have that for any $T \ge 1$, the following holds

$$f(0) - \inf_x f(x) + \frac{\|d_0\|^2}{MU} \le \frac{150\pi\epsilon}{U} \times (10\pi T + 0.03) \le \frac{150\pi\epsilon}{U} \times 11\pi T.$$

Property 2) is true due to the definition of $\bar{f}$.
Property 3) is true because the following

$$\|\nabla\bar{f}(z) - \nabla\bar{f}(y)\| = \sqrt{2\epsilon}\left\|\nabla\bar{h}\left(\frac{zU}{75\pi\sqrt{2\epsilon}}\right) - \nabla\bar{h}\left(\frac{yU}{75\pi\sqrt{2\epsilon}}\right)\right\| \le U\|z - y\|$$

where the last inequality comes from Lemma 3.1 – (5). This completes the proof. **Q.E.D.**

Next let us analyze the size of $\nabla\bar{h}$. We have the following result.

17

**Lemma 3.3** *If there exists $k \in [T]$ such that $|y[k]| < 1$, then*

$$\left\| \nabla \bar{h}(y) \right\| = \left\| \frac{1}{M} \sum_{i=1}^{M} \nabla h_i(y) \right\| \geq \left| \frac{1}{M} \sum_{i=1}^{M} \frac{\partial}{\partial y[k]} h_i(y) \right| > 1.$$

**Proof.** The first inequality holds for all $k \in [T]$, since $\frac{1}{M} \sum_{i=1}^{M} \frac{\partial}{\partial y[k]} h_i(y)$ is one element of $\frac{1}{M} \sum_{i=1}^{M} \nabla h_i(y)$.

We divide the proof for second inequality into two cases.

**Case 1.** Suppose $|y[j-1]| < 1$ for all $2 \leq j \leq k$. Therefore, we have $|y[1]| < 1$. Using (44), we have the following inequalities:

$$\frac{\partial}{\partial y[1]} h_i(y) \overset{(i)}{\leq} -\Psi(1)\Phi'(y[1]) \overset{(ii)}{<} -1, \forall i \tag{51}$$

where (i) is true because $\Psi'(w), \Phi(w)$ are all non-negative from Lemma 3.1 -(2); (ii) is true due to Lemma 3.1 – (3). Therefore, we have the following

$$\left\| \nabla \bar{h}(y) \right\| = \left\| \frac{1}{M} \sum_{i=1}^{M} \nabla h_i(y) \right\| \geq \left| \frac{1}{M} \sum_{i=1}^{M} \frac{\partial}{\partial y[1]} h_i(y) \right| > 1.$$

**Case 2)** Suppose there exists $2 \leq j \leq k$ such that $|y[j-1]| \geq 1$.

We choose $j$ so that $|y[j-1]| \geq 1$ and $|y[j]| < 1$. Therefore, depending on the choices of $(i, j)$ we have three cases

$$\frac{\partial h_i(y)}{\partial y[j]} = \begin{cases} -3\left(\Psi(-y[i-1])\Phi'(-y[j]) + \Psi(y[i-1])\Phi'(y[j])\right), & i \in [1, \frac{M}{3}] \\ 0, & i \in [\frac{M}{3}+1, \frac{2M}{3}] \\ -3\left(\Psi'(-y[j])\Phi(-y[i+1]) + \Psi'(y[j])\Phi(y[i+1])\right), & i \in [\frac{2M}{3}+1, M] \end{cases}.$$

If $i \in [1, \frac{M}{3}]$, because $|y[j-1]| \geq 1$ and $|y[j]| < 1$, using Lemma 3.1 – (3), and the fact that the negative part is zero for $\Psi$, and $\Phi'$ is even function, the expression further equals to

$$-3 \cdot \Psi(|y[j-1]|)\Phi'(|y[j]|) \overset{(37)}{<} -3, \tag{52}$$

If $i \in [\frac{2M}{3}+1, M]$ the expression is obviously non-positive because both $\Psi'$ and $\Phi$ are nonnegative. Overall, we have

$$\left| \frac{1}{M} \sum_{i=1}^{M} \frac{\partial h_i(y)}{\partial y[j]} \right| > \left| \frac{1}{M} \sum_{i=1}^{M/3} 3 \right| = 1.$$

This completes the proof. **Q.E.D.**

**Lemma 3.4** *Define $\bar{x} := \frac{1}{M} \sum_{i=1}^{M} x_i$, and assume that $U \in (0, 1)$. Then we have*

$$\left\| \frac{1}{M} \sum_{i=1}^{M} \nabla f_i(x_i) \right\|^2 + \frac{U}{M\underline{\lambda}_{\min}(P^{1/2}\mathcal{L}P^{1/2})} \sum_{(i,j):i\sim j} \|x_i - x_j\|^2 \geq \frac{1}{2} \left\| \nabla \bar{f}(\bar{x}) \right\|^2.$$

**Proof.** First let us derive a useful property. Define $d := [d_1; d_2; \cdots ; d_M]$ where $d_i$ is the degree for node $i$;

18

further define

$$\bar{x} := \frac{1}{M} \sum_{i=1}^{M} x_i, \quad \tilde{x}_i := x_i - \bar{x}, \quad \tilde{x} := [\tilde{x}_1; \tilde{x}_2; \cdots ; \tilde{x}_M].$$

It is easy to observe that :

$$\tilde{x}^T \mathbb{1} = 0, \quad \text{and} \quad \tilde{x} \notin \text{Null}(F^T F).$$

Then the following holds:

$$x^T F^T F x = \sum_{(i,j):i\sim j} \|x_i - x_j\|^2 = \sum_{(i,j):i\sim j} \|\tilde{x}_i - \tilde{x}_j\|^2 = \tilde{x}^T F^T F \tilde{x} \geq \underline{\lambda}_{\min}(F^T F) \|\tilde{x}\|^2. \tag{53}$$

Therefore the following holds:

$$\sum_{i=1}^{M} \|\bar{x} - x_i\|^2 \leq \frac{1}{\underline{\lambda}_{\min}(F^T F)} \sum_{(i,j):i\sim j} \|x_i - x_j\|^2 = \frac{1}{\underline{\lambda}_{\min}(P^{1/2}\mathcal{L}P^{1/2})} \sum_{(i,j):i\sim j} \|x_i - x_j\|^2. \tag{54}$$

Based on the above property, we have the following series of inequalities

$$\left\| \nabla \bar{f}(\bar{x}) \right\|^2 \leq 2 \left\| \frac{1}{M} \sum_{i=1}^{M} (\nabla f_i(\bar{x}) - \nabla f_i(x_i)) \right\|^2 + 2 \left\| \frac{1}{M} \sum_{i=1}^{M} \nabla f_i(x_i) \right\|^2$$

$$\overset{(i)}{\leq} \frac{2}{M} \sum_{i=1}^{M} \left\| \nabla f_i(\frac{1}{M} \sum_{j=1}^{M} x_j) - \nabla f_i(x_i) \right\|^2 + 2 \left\| \frac{1}{M} \sum_{i=1}^{M} \nabla f_i(x_i) \right\|^2$$

$$\overset{(ii)}{\leq} \frac{2}{M} \sum_{i=1}^{M} U^2 \left\| \frac{1}{M} \sum_{j=1}^{M} x_j - x_i \right\|^2 + 2 \left\| \frac{1}{M} \sum_{i=1}^{M} \nabla f_i(x_i) \right\|^2$$

$$\overset{(iii)}{\leq} \frac{2U}{M\underline{\lambda}_{\min}(P^{1/2}\mathcal{L}P^{1/2})} \sum_{(i,j):i\sim j} \|x_j - x_i\|^2 + 2 \left\| \frac{1}{M} \sum_{i=1}^{M} \nabla f_i(x_i) \right\|^2,$$

where in (i) and (iii) we have used the convexity of the function $\| \cdot \|^2$; in (ii) we used Lemma 3.2 – (3); in (iii) we have also used the assumption that $U \in (0, 1)$ and (54). Overall we have

$$\left\| \frac{1}{M} \sum_{i=1}^{M} \nabla f_i(x_i) \right\|^2 + \frac{U}{M\underline{\lambda}_{\min}(P^{1/2}\mathcal{L}P^{1/2})} \sum_{(i,j):i\sim j} \|x_i - x_j\|^2 \geq \frac{1}{2} \|\nabla f(\bar{x})\|^2 .$$

This completes the proof. **Q.E.D.**

**Lemma 3.5** *Consider using an algorithm in class $\mathcal{A}$ or in class $\mathcal{A}'$ to solve the following problem:*

$$\min_{x\in\mathbb{R}^{TM\times 1}} \quad h(x) = \frac{1}{M} \sum_{i=1}^{M} h_i(x_i), \tag{55}$$

*over a path graph. Assume the initial solution: $x_i = 0$, $\forall \ i \in [M]$. Let $\bar{x} = \frac{1}{M} \sum_{i=1}^{M} x_i$ denote the average of the local variables. Then the algorithm needs at least $(\frac{M}{3} + 1)T$ iterations to have $x_i[T] \neq 0$, $\forall \ i$ and*

19

$\bar{x}[T] \neq 0$.

**Proof.** For a given $k \geq 2$, suppose that $x_i[k], x_i[k+1], ..., x_i[T] = 0$, $\forall i$, that is, support$\{x_i\} \subseteq \{1, 2, 3, ..., k-1\}$ for all $i$. Then $\Psi'(x_i[k]) = \Psi'(-x_i[k]) = 0$ for all $i$, and $h_i$ has the following partial derivative when $k$ is even:

$$\frac{\partial h_i(x_i)}{\partial x_i[k]} = \begin{cases} -3 \left( \Psi\left(-x_i[k-1]\right) \Phi'\left(-x_i[k]\right)\right) + 3 \left( \Psi\left(x_i[k-1]\right) \Phi'\left(x_i[k]\right)\right), & i \in [1, \frac{M}{3}] \\ 0, & i \in [\frac{M}{3}+1, M] \end{cases} \quad (56)$$

and the following partial derivative when $k$ is odd and $k \geq 3$:

$$\frac{\partial h_i(x_i)}{\partial x_i[k]} = \begin{cases} 0, & i \in [1, \frac{2M}{3}] \\ -3 \left( \Psi\left(-x_i[k-1]\right) \Phi'\left(-x_i[k]\right)\right) + 3 \left( \Psi\left(x_i[k-1]\right) \Phi'\left(x_i[k]\right)\right), & i \in [\frac{2M}{3}+1, M] \end{cases} . \quad (57)$$

Recall that for any algorithm in class $\mathcal{A}$ or $\mathcal{A}'$, each agent is only able to compute linear combination of historical gradient and neighboring iterates [cf. (15) and (16)]. Therefore, for a given node $i$, as long as the $k$th element of the gradient as well as that of its neighbors have never been updated once, $x_i[k]$ remains to be zero. Combining this observation with the above two expressions for $\frac{\partial h_i(x_i)}{\partial x_i[k]}$, we can conclude that when support$\{x_i\} \subseteq \{1, 2, 3, ..., k-1\}$ for all $i$, then in the next iteration $x_i[k]$ will be possibly non-zero on the node $i \in [1, \frac{M}{3}]$ for even $k$ and $i \in [\frac{2M}{3}+1, M]$ for odd $k$, and all other nodes still have $x_j[k] = 0$, $\forall j \neq i$.

Now suppose that the initial solution is $x_i[k] = 0$ for all $(i, k)$. Then at the first iteration only $\frac{\partial h_i(x_i)}{\partial x_i[1]}$ is non-zero for all $i$, due to the fact that $\frac{\partial h_i(x_i)}{\partial x_i[1]} = \Psi(1)\Phi'(0) = 4(1 - e^{-1})$ for all $i$ from (44). If follows that even if every node is able to compute its local gradient, and can communicate with their neighbors, it is only possible to have $x_i[1] \neq 0, \forall i$. At the second iteration, we can use (56) to conclude that it is only possible to have $\frac{\partial h_r(x_r)}{\partial x_r[k]} \neq 0$ for some $r \in [1, M/3]$, therefore when using an algorithm in class $\mathcal{A}$, we can conclude that $x_i[2] = 0$ for all $i \notin [1, M/3]$.

Then following our construction (33), we know the nodes in the set $[1, \frac{M}{3}]$ and the set $[\frac{2M}{3}+1, M]$ have minimum distance $M/3$. It follows that using an algorithm in $\mathcal{A}$ or $\mathcal{A}'$, it takes at least $M/3$ iterations for the non-zero $x_r[2]$ and the corresponding gradient vector to propagate to at least one node in set $[2M/3+1, M]$. Once we have $x_j[2] \neq 0$ for some $j \in [2M/3+1, M]$, then according to (57), it is possible to have $\frac{\partial h_j(x_j)}{\partial x_j[3]} \neq 0$, and once this gradient becomes non-zero, the corresponding variable $x_j[3], j \in [2M/3+1, M]$ can become nonzero.

Following the above procedure, it is clear that we need at least $\frac{MT}{3}$ iterates and $T$ computations to make $x_i[T]$ possibly non-zero. **Q.E.D.**

**Theorem 3.1** *Let $U \in (0, 1)$ and $\epsilon$ be positive. Let $x^0[i] = 0$ for all $i \in [M]$. Then for any distributed first-order algorithm in class $\mathcal{A}$ or $\mathcal{A}'$, there exists a problem in class $\mathcal{P}_U^M$ and a network in class $\mathcal{N}$, such that it requires at least the following number of iterations*

$$t \geq \frac{1}{3\sqrt{\xi(\mathcal{G})}} \left\lfloor \frac{\left( f(0) - \inf_x f(x) + \frac{\|d_0\|^2}{MU} \right) U}{1650\pi^2} \epsilon^{-1} \right\rfloor \quad (58)$$

20

*to achieve the following error*

$$h_t^* = \left\| \frac{1}{M} \sum_{i=1}^{M} \nabla f_i(x_i^t) \right\|^2 + \frac{U}{M \underline{\lambda}_{\min}(P^{1/2} \mathcal{L} P^{1/2})} \sum_{(i,j):i \sim j} \|x_i^t - x_j^t\|^2 < \epsilon. \tag{59}$$

**Proof.** By Lemma 3.5 we have $\bar{x}[T] = 0$ for all $t < \frac{M+3}{3}T$. Then by applying Lemma 3.2 – (2) and Lemma 3.3, we can conclude that the following holds

$$\left\| \nabla \bar{f}(\bar{x}[T]) \right\| = \sqrt{2\epsilon} \left\| \nabla \bar{h} \left( \frac{\bar{x}[T]U}{75\pi\sqrt{2\epsilon}} \right) \right\| > \sqrt{2\epsilon}, \tag{60}$$

where the second inequality follows that there exists $k \in [T]$ such that $|\frac{\bar{x}[k]U}{75\pi\sqrt{2\epsilon}}| = 0 < 1$, then we can directly apply Lemma 3.3. Then by applying Lemma 3.4 gives $h_{(M+3)T/3}^* > \epsilon$.

The third part of Lemma 3.2 ensures that $f_i$'s are $U$-Lipschitz continuous gradient, and the first part shows

$$f(0) - \inf_x f(x) + \frac{\|d_0\|^2}{MU} \leq \frac{1650\pi^2\epsilon}{U}T,$$

Therefore we obtain

$$T \geq \left\lfloor \frac{\left( f(0) - \inf_x f(x) + \frac{\|d_0\|^2}{MU} \right) U}{1650\pi^2} \epsilon^{-1} \right\rfloor. \tag{61}$$

Summarizing the above argument, we have

$$t \geq \frac{M+3}{3}T \geq \frac{M+3}{3} \left\lfloor \frac{\left( f(0) - \inf_x f(x) + \frac{\|d_0\|^2}{MU} \right) U}{1650\pi^2} \epsilon^{-1} \right\rfloor.$$

By noting that for path graph $\xi(\mathcal{G}) \geq 1/M^2$, this completes the proof. **Q.E.D.**

## 3.2 Generalization

The previous section analyzes the lower complexity bounds for problem $\mathcal{P}_U^M$ over a path network. The obtained results can be extended in a number of direction.

### 3.2.1 Uniform $L_i$, Fixed $D$ and $M$

In this subsection, we would like to generalize Theorem 3.1 to a slightly wider class of networks (beyond the path graph used in our construction). Towards this end, consider a *path-star graph* shown in Fig. 5. The graph contains a path graph with $D-1$ nodes, and the remaining nodes are divided into $D-1$ groups, each with either $\lfloor M/(D-1) - 1 \rfloor$ or $\lfloor M/(D-1) - 1 \rfloor + 1$ nodes, and each group is connected to the nodes in the path graph by using a star topology. We have the following corollary to Theorem 3.1.

**Corollary 3.1** *Let $U \in (0,1)$ and $\epsilon$ be positive, and fix any $D$ and $M$ such that $D \leq M - 1$. For any algorithm in class $\mathcal{A}$ or $\mathcal{A}'$, there exists a problem in class $\mathcal{P}_U^M$ and a network in class $\mathcal{N}_D^M$, so that to*

**Figure 5:** The path-star graph used in our construction.

*achieve accuracy $h_t^* < \epsilon$, it requires at least the following number iterations*

$$t \geq \frac{D}{3} \left\lfloor \frac{\left( f(0) - \inf_x f(x) + \frac{\|d_0\|^2}{MU} \right) U}{1650\pi^2} \epsilon^{-1} \right\rfloor .$$

*Alternatively, the above bound can be expressed as the following*

$$t \geq \frac{\sqrt{(D-1)/(2M)}}{3\sqrt{\xi(\mathcal{G})}} \left\lfloor \frac{\left( f(0) - \inf_x f(x) + \frac{\|d_0\|^2}{MU} \right) U}{1650\pi^2} \epsilon^{-1} \right\rfloor .$$

**Proof.** Fix any $D$ and $M$ such that $D \leq M - 1$, we can construct a path-star graph as described in Fig.5, whose diameter is $D$.

To show the lower bounds for such a graph, we split all $M$ nodes into three sets $\mathcal{A}, \mathcal{B}, \mathcal{C}$ based on the main path, each with $\frac{M}{3}$ nodes (assume $M$ is a multiple of 3), where $\mathcal{A}$ and $\mathcal{C}$ has minimum $\frac{D+2}{3}$ distance (assume $D - 1$ is a multiple of 3). Then we construct the component functions $h_i$'s as follows.

$$h_i(x_i) = \begin{cases} \Theta(x_i, 1) + 3 \sum_{j=1}^{\lfloor T/2 \rfloor} \Theta(x_i, 2j), & i \in \mathcal{A} \\ \Theta(x_i, 1), & i \in \mathcal{B} \\ \Theta(x_i, 1) + 3 \sum_{j=1}^{\lfloor T/2 \rfloor} \Theta(x_i, 2j+1), & i \in \mathcal{C} \end{cases} \tag{62}$$

Since the graph has diameter $D$ in the above construction, and the distance between any two elements in $\mathcal{A}$ and $\mathcal{C}$ is at least $\frac{D+2}{3}$ (assume $D - 1$ is a multiple of 3), by a similar step in Lemma 3.5 we can conclude that we need at least $(\frac{D+2}{3}+1)T$ iterations to achieve $x_i[T] \neq 0$. By applying (61), we can obtain the desired result.

22

To show the second result, note that from (28) we have

$$\sum_i d_i D \geq \frac{1}{\underline{\lambda}_{\min}(\mathcal{L})} \tag{63}$$

For the path-star graph under consideration,

$$\sum_i d_i \leq 2(D-1) - 2 + 2\left(M - (D-1)\right) \leq 2M,$$

so the following holds:

$$D^2 \geq \frac{D/2M}{\underline{\lambda}_{\min}(\mathcal{L})} \geq \frac{(D-1)/(2M)}{\underline{\lambda}_{\min}(\mathcal{L})}.$$

The desired result is then immediate.                                               **Q.E.D.**

### 3.2.2  Non-uniform $L_i$, Fixed $\mathcal{N}$

Finally, for the problem class with non-uniform Lipschitz constants, we can extend the previous result to any network in class $\mathcal{N}$ (by properly assigning different values of $L_i$'s to different nodes). In this case the lower bound will be dependent on the spectrum property of $\widehat{\mathcal{L}}$ as defined in (26) (expressed below for easy reference)

$$\widehat{\mathcal{L}} := L^{-1/2} F^T K F L^{-1/2}. \tag{64}$$

**Corollary 3.2** *Let $\epsilon$ be positive. For any given network in $\mathcal{N}_D^M$, and for any algorithm in $\mathcal{A}$, there exists a problem in $\mathcal{P}_L^M$ such that to achieve accuracy $h_t^* < \epsilon$, it requires at least the following iterations*

$$t \geq \frac{1}{3\sqrt{\xi(\widehat{\mathcal{L}})}} \left\lfloor \frac{\left(f(0) - \inf_x f(x) + \frac{\|d_0\|_{L^{-1}}^2}{M}\right)\bar{L}}{1650\pi^2} \epsilon^{-1} \right\rfloor. \tag{65}$$

To prove this result, we select the values of the coefficient set $\{L_i\}_{i=1}^M$, so that the "effective" network topology becomes a path. In particular, for any given network in $\mathcal{N}$, we can construct local functions as follows: First, along the longest path of size $D$, we distributed the functions into three sets $\mathcal{A}, \mathcal{B}, \mathcal{C}$, where $\mathcal{A}$ and $\mathcal{C}$ denotes the first and last $\frac{D}{3}$ nodes on the path respectively, and $\mathcal{B}$ denotes the rest nodes on the path. Second, for the rest of the functions not on the path, denoted as set $\mathcal{D}$, set their local functions to zero (or equivalently, set the corresponding $L_i$'s to zero). Then, the local function belongs to each set can be expressed as:

$$
h_i(x_i) = \begin{cases}
\dfrac{M}{D}\Theta(x_i,1) + \dfrac{3M}{D}\displaystyle\sum_{j=1}^{\lfloor T/2 \rfloor}\Theta(x_i,2j), & i \in \mathcal{A} \\[1.8em]
\dfrac{M}{D}\Theta(x_i,1), & i \in \mathcal{B} \\[1.8em]
\dfrac{M}{D}\Theta(x_i,1) + \dfrac{3M}{D}\displaystyle\sum_{j=1}^{\lfloor T/2 \rfloor}\Theta(x_i,2j+1), & i \in \mathcal{C} \\[1.8em]
0, & i \in \mathcal{D}
\end{cases}
\tag{66}
$$

This way the network reduces to a path graph. Note that the Lipschitz constant for the gradient of $h(y) = \frac{1}{M}\sum_{i=1}^{M} h_i(y)$ is still 1, and we can use the similar constructions and proof steps leading to Theorem 3.1 to prove the claim.

# 4    The Proposed Algorithms

In this section, we introduce two different types of algorithms for solving problem (2). The algorithm is *near-optimal*, and can achieve the lower bounds derived in Section 3 except for a multiplicative polylog factor in $M$. To simplify the notation, we utilize the definitions introduced in Section 2, and rewrite problem (2) in the following compact form

$$
\min_{x \in \mathbb{R}^{SM}} \ f(x) := \frac{1}{M}\sum_{i=1}^{M} f_i(x_i), \quad \text{s.t. } (F \otimes I_S)x = 0.
\tag{67}
$$

It can be verified that, by using the definition of $F$, the constraint in this problem is equivalent to the ones given in (2). For notational simplicity, in the following we will assume that $S = 1$ (scalar variables). All the results presented in subsequent sections extend easily to case with $S > 1$.

## 4.1    The D-GPDA Algorithm

We first present a Distributed Gradient Primal-Dual Algorithm (D-GPDA), which relaxes the linear constraint (67), and gradually enforces it as the algorithm proceeds. To describe the algorithm, let us introduce the augmented Lagrangian (AL) function as

$$
\mathsf{AL}(x,\lambda) = f(x) + \langle \lambda, Fx \rangle + \frac{1}{2}\|\Sigma Fx\|^2,
\tag{68}
$$

where $\lambda \in \mathbb{R}^E$ is the dual variable; $\Sigma = \text{diag}([\sigma_1,\cdots,\sigma_E]) \in \mathbb{R}^{E \times E}$ is a diagonal positive definite matrix. In the following, we will use the shorthanded notation $\mathsf{AL}^r := \mathsf{AL}(x^r,\lambda^r)$ where $r$ is the iteration counter.

Define a *penalty matrix* as

$$
\Upsilon = \text{diag}\{[\beta_1,\cdots,\beta_M]\} \succ 0.
\tag{69}
$$

Then the D-GPDA is described in the following table.

**Algorithm 1. The D-GPDA Algorithm**

**(S1).** Assign each node $i \in \mathcal{N}$ with a parameter $\beta_i > 0$; Assign each edge $(ij) \in \mathcal{E}$ with a parameter $\sigma_{ij} > 0$;

**(S2).** At iteration $r = -1$, initialize $\lambda^{-1} = 0$ and $x^{-1} = 0$;

**(S3).** At iteration $r = 0$, set $\lambda^0$ and $x^0$ using the following:

$$\nabla f(x^{-1}) + (2\Delta + \Upsilon^2)x^0 = 0, \ \lambda^0 = \Sigma^2 Fx^0; \tag{70}$$

Equivalently $x^0$ can be written as:

$$x_i^0 = \left( 2 \sum_{j:j\sim i} \sigma_{ij}^2 + \beta_i^2 \right)^{-1} \nabla f_i(0)/M, \ \forall \ i \in [M]; \tag{71}$$

**(S4).** At each iteration $r + 1$, $r \geq 0$, update variables by:

$$x^{r+1} = \arg\min_x \langle \nabla f(x^r) + F^T \lambda^r, x - x^r \rangle \tag{72a}$$

$$+ \frac{1}{2}\|\Sigma Fx\|^2 + \frac{1}{2}\|\Sigma B(x - x^r)\|^2 + \frac{1}{2}\|\Upsilon(x - x^r)\|^2$$

$$\lambda^{r+1} = \lambda^r + \Sigma^2 Fx^{r+1}. \tag{72b}$$

We note that each iteration of the D-GPDA performs a gradient descent type step on the AL function, followed by taking a step of dual gradient ascent (with a *stepsize matrix* $\Sigma^2 \succ 0$). The term $\frac{1}{2}\|\Sigma B(x-x^r)\|^2$ used in (72a) is a *network proximal term* that regularizes the $x$ update using network structure, and its presence is critical to ensure separability and distributed implementation (see Remark 4.3 below).

The D-GPDA is closely related to many classical primal-dual methods, such as the Uzawa method [48] (which has been recently utilized to solve linearly constrained *convex* problems [49]), and the proximal method of multipliers (prox-MM) [50, 51]. The latter method has been first developed by Rockafellar in [50], in which a proximal term has been added to the AL in order to make it strongly convex in each iteration. However, the theoretical results derived for Prox-MM in [50, 51] are only valid for convex problems. It is also important to note that when the matrices $\Sigma$ and $\Upsilon$ are specialized as multiples of identity matrices, that is, when $\Sigma = \sigma I_M$ and $\Upsilon = \kappa I_M$ for some $\sigma, \kappa > 0$, then the D-GPDA reduces to the Prox-GPDA algorithm briefly discussed in our earlier work [15, Section 5], for solving a general linearly constrained problem.

## 4.2 The xFILTER Algorithm

Despite the fact that D-GPDA is conceptually simple, we will show shortly that it is only optimal for special network classes with small diameter [or large gap function $\xi(\mathcal{G})$], such as the complete/star networks (see Table 1 and our detailed analysis in Section 6). Intuitively, the issue is that having the *network proximal term* imposes very heavy regularization, enforcing the new iterates to be close to the old ones. This causes slow information propagation over the network.

In this section, we present a *near-optimal* algorithm that can achieve the lower bounds derived in Section 3 for a number of different graphs (up to some polylog factor in the problem dimension). To motivate our algorithm design, observe that the communication lower bound $\mathcal{O}(1/\sqrt{\xi(\mathcal{G})} \times \bar{L}/\epsilon)$ in Section 3 can be decomposed into the product two parts, $\mathcal{O}(1/\sqrt{\xi(\mathcal{G})})$ and $\mathcal{O}(\bar{L}/\epsilon)$, corresponding roughly to the

communication efficiency and the computational complexity, respectively. Such a product form motivates us to *separate* the computation and communication tasks, and design a *double loop* algorithm to achieve the desired lower bound.

Our proposed algorithm is based on a novel *approximate filtering -then- predict and tracking* (xFILTER) strategy, which properly combines the modern first-order optimization methods and the classical polynomial filtering techniques. It is a "double-loop" algorithm, where in the outer loop local gradients are computed to extract information from local functions, while in the inner loop some filtering techniques are used to facilitate efficient information propagation. Please see **Algorithm 1** for the detailed description, from the system perspective. It is important to note that the algorithm contains an outer loop **(S3)**–**(S4)** and an inner loop **(S2)**, indexed by $r$ and $q$, respectively. Further, the local gradient evaluation only appears in the outer loop step **(S3)**.

To understand the algorithm, we note that one important task of each agent is to update its local variable so that it is close to the average $\frac{1}{M}\sum_{i=1}^{M} x_i$. Let us use $d_i$ to denote a local variable that *approximates* the above average. At the beginning of the algorithm, $d_i$ is just a rough estimate of the average, so we have $d_i = \frac{1}{M}\sum_j x_j + e_i$, where $e_i$ is the *deviation* from the true average, and it can be viewed as some kind of "estimation noise". To gradually remove such a noise, in step **S1)** we resort to the so-called graph based joint bilateral filtering used for image denoising [52, 53], which can be formulated as the following regularized least squares problem:

$$x_*^{r+1} := \arg\min_{x\in\mathbb{R}^M} \frac{1}{2}\|x - d^r\|_{\Upsilon^2}^2 + \frac{1}{2}x^\top F^\top \Sigma^2 F x, \tag{73}$$

where $d^r$ is the noisy signal, $F$ is a penalty high pass filter related to the graph structure (in our case, $F$ is the adjacency matrix), and $\Sigma^2$ is a regularization parameter. Its solution, denoted as $x_*^{r+1}$ as given below, will be close to the "unfiltered" signal $d^r$, while having reduced high frequency components, or high fluctuations across the components:

$$R x_*^{r+1} = d^r, \quad \text{with} \quad R := \Upsilon^{-2} F^\top \Sigma^2 F + I_M. \tag{74}$$

It is important to note that if $x_*^{r+1}$ indeed achieves consensus, then by (11) we have $F^\top \Sigma^2 F x_*^{r+1} = 0$, implying $x_*^{r+1} = d^r$, which says $d^r$ should "track" $x_*^{r+1}$.

Unfortunately, the system (74) cannot be precisely solved in a distributed manner, because inverting $R$ destroys its pattern about the network structure embedded in the product $F^\top \Sigma^2 F$. More specifically, $F^\top \Sigma^2 F$ is the weighted graph Laplacian matrix whose $(i,j)$th entry is nonzero if and only if node $i, j$ are connected, but $(\Upsilon^{-2} F^\top \Sigma^2 F + I_M)^{-1}$ is a dense matrix without such a property. Therefore in **S2)**, we use a degree-$Q$ Chebyshev polynomial to approximate $x_*^{r+1}$. The output, denoted as $x^{r+1}$, stays in a Krylov space $span\{d^r, Rd^r, \cdots, R^Q d^r\}$. Specifically, at each iteration, the only step that requires communication is the operation $Ru$, which is given by

$$(Ru_{q-1})[i] = (\Upsilon^{-2} F^\top \Sigma^2 F u_{q-1})[i] + d_{q-1}[i] \tag{75}$$
$$= \frac{1}{\beta_i^2} \sum_{j:j\sim i} \sigma_{ij}^2 (u_{q-1}[j] - d^r[i]) + u_{q-1}[i], \ \forall \ i,$$

so this step can be done distributedly, via one round of local message exchange.

After completing $Q > 0$ such Chebyshev iterations (77) (C-iteration for short), the obtained solution

$x^{r+1}$ will be an approximate solution to the system 74, with a residual error vector $\epsilon^{r+1}$ as given below

$$Rx^{r+1} = d^r + R\epsilon^{r+1}, \quad \text{with} \quad \epsilon^{r+1} := x^{r+1} - x_*^{r+1}. \tag{76}$$

Up to this point, the filtering technique we have discussed aims at removing the "non-consensus" parts from a vector $d = [d_1, \cdots, d_N]^T$. However, recall that the goal of distributed optimization is not only to achieve consensus, but also to optimize the objective function $\sum_i f_i(x_i)$. Therefore, a *prediction* step **(S3)** is performed to incorporate the most up-to-date local gradient $\nabla f_i(x_i)$. Then a *tracking* step **(S4)** is performed to update $d$. Ideally, one would like the new $d_i^{r+1}$ to have the following three properties: 1) It is close to the previous $d_i^r$; 2) it takes into consideration the new local gradient information offered by the "predicted" $\tilde{x}_i^{r+1}$; 3) it is a "low frequency" signal, meaning $d_i^{r+1}$ and $d_j^{r+1}$ are relatively close, for all $i \neq j$. Taking a closer look at the "tracking" step, we can see that all three components are included: It adds to the previous $d^r$ the differences of the last two predictions, and it removes some non-consensus components among the local variables. The detailed algorithm is given in the Algorithm 2.

To end this subsection, we emphasize that, the use of the polynomial Chebyshev filtering requires $Q$ vector communications steps every time that **(S2)** is performed. However, such a filtering step is critical to make the proposed algorithm achieve performance lower bounds predicted in Section 3. Intuitively, it helps to accelerate information propagation across the network. Indeed, as will be shown shortly, the number $Q$ in **(S2)** is directly related to properties of the underlying graph. It is also somewhat surprising that the inner problem (73) is not required to be solved with *increased* accuracy. On the contrary, only a *fixed* number of filtering steps are needed.

---

**Algorithm 2. The xFILTER Algorithm**

**(S1) [Initialization].** Assign each node $i \in \mathcal{N}$ with $\beta_i > 0$; Assign each edge $(ij) \in \mathcal{E}$ with $\sigma_{ij} > 0$; Initialize $x^{-1} = 0$, $d^{-1} = -\Upsilon^{-2}\nabla f(x^{-1})$ and $\tilde{x}^{-1} = x^{-1} - \Upsilon^{-2}\nabla f(x^{-1})$. Compute $R$ by (74);

**(S2) [Filtering].** At iteration $r + 1$, $r \geq -1$: For a fixed constant $Q > 0$, run the following C-iterations (with parameters $\{\alpha_q, \tau\}$)

$$u_0 = x^r, \ u_1 = (I - \tau R)u_0 + \tau d^r, \tag{77}$$
$$u_q = \alpha_q(I - \tau R)u_{q-1} + (1 - \alpha_q)u_{q-2} + \tau\alpha_q d^r, \ q = 2, \cdots, Q;$$

Set $x^{r+1} = u_Q$;

**(S3) [Prediction].** Compute $\tilde{x}^{r+1}$ by:

$$\tilde{x}^{r+1} = x^{r+1} - \Upsilon^{-2}\nabla f(x^{r+1}); \tag{78}$$

**(S4) [Tracking].** Compute $d^{r+1}$ by:

$$d^{r+1} = d^r + (\tilde{x}^{r+1} - \tilde{x}^r) - \Upsilon^{-2}F^\top\Sigma^2 Fx^{r+1}. \tag{79}$$

Set $r = r + 1$, go to **(S2)**.

---

## 4.3 Discussion

In this subsection, we establish some key connections between the two algorithms discussed so far, and provide some additional remarks.

First, we provide an important interpretation of the xFILTER strategy, which will help us subsequently provide an unified analysis framework for both D-GPDA and xFILTER. First, similarly as in the D-GPDA algorithm, let us introduce an auxiliary variable $\lambda^r \in \mathbb{R}^E$, which is updated as follows:

$$\lambda^{r+1} = \lambda^r + \Sigma^2 F x^{r+1}. \tag{80}$$

Suppose $\lambda^{-1} = 0$, then according to (79) and (78) we have the following relationship

$$d^0 := -\Upsilon^{-2}\nabla f(x^{-1}) + (x^0 - \Upsilon^{-2}\nabla f(x^0) - (x^{-1} - \Upsilon^{-2}\nabla f(x^{-1}))) - \Upsilon^{-2}F^T\lambda^0$$
$$= x^0 - \Upsilon^{-2}\nabla f(x^0) - \Upsilon^{-2}F^T\lambda^0.$$

By using the induction argument, we can show that for all $r \geq 0$, the following holds

$$d^r := x^r - \Upsilon^{-2}\nabla f(x^r) - \Upsilon^{-2}F^T\lambda^r. \tag{81}$$

Combining (74) and (81), we obtain the following useful alternative expressions of (74) and (76):

$$\Upsilon^{-2}\Big(\nabla f(x^r) + F^\top(\lambda^r + \Sigma^2 F x_*^{r+1})\Big) + (x_*^{r+1} - x^r) = 0 \tag{82a}$$

$$\Upsilon^{-2}\Big(\nabla f(x^r) + F^\top(\lambda^r + \Sigma^2 F x^{r+1})\Big) + (x^{r+1} - x^r) = R\epsilon^{r+1}. \tag{82b}$$

Using (82a), it is clear that $x_*^{r+1}$ can be equivalently written as the optimal solution of the following problem:

$$x_*^{r+1} = \operatorname*{argmin}_x \langle \nabla f(x^r) + F^T\lambda^r, x - x^r \rangle + \frac{1}{2}\|\Sigma F x\|^2 + \frac{1}{2}\|\Upsilon(x - x^r)\|^2. \tag{83}$$

The relations (80) and (83) together show that D-GPDA and xFILTER are closely related. However, we note that when comparing (83) with (72a), one key difference is that the *network proximal* term $\frac{1}{2}\|\Sigma B(x - x^r)\|^2$ used in D-GPDA is no longer used in xFILTER.

We have the following additional remarks on the proposed algorithms.

**Remark 4.1 (Parameters)** *It is important to note that in both Alg. 1 and 2, in the update of the primal and dual variables, some "matrix parameters" are used instead of scalar ones. In particular, the matrix $\Upsilon^2$ is used as the primal "proximal parameter", while $\Sigma^2$ is used as the "dual stepsize". Using these matrices ensures that we can appropriately design parameters for each node/link, which is one key ingredient in ensuring the optimal rate.*

**Remark 4.2 (Initialization)** *The initialization steps in (S2) and (S3) of Alg. 1 can be done in a distributed manner. Each node $i$ only requires to know the neighbors' $\sigma_{ij}^2$'s in order to update $x_i^0$. Once $x^0$ is updated, $\lambda^0$ can be updated by using:*

$$\lambda_{ij}^0 = \sigma_{ij}^2(x_i^0 - x_j^0), \quad \forall\,(i,j) \in E.$$

**Remark 4.3 (Distributed Implementation and Algorithm Classes)** *To see how the D-GPDA can be executed distributedly, we write down the optimality condition of (72a). For notational simplicity, define:*

$$H := B^T \Sigma^2 B + \Upsilon^2. \tag{84}$$

*Then we have*

$$\nabla f(x^r) + F^T \lambda^r + F^T \Sigma^2 F x^{r+1} + H(x^{r+1} - x^r) = 0. \tag{85}$$

*Rearranging, and using property (22), we have*

$$\nabla f(x^r) + F^T \lambda^r + (2\Delta + \Upsilon^2) x^{r+1} - H x^r = 0.$$

*Subtracting the same equation from the rth iteration, and use the fact that $F^T(\lambda^r - \lambda^{r-1}) = F^T \Sigma^2 F x^r$, we have*

$$x^{r+1} = x^r - \left(2\Delta + \Upsilon^2\right)^{-1} \left(\nabla f(x^r) - \nabla f(x^{r-1}) + (F^T \Sigma^2 F - H)x^r + H x^{r-1}\right). \tag{86}$$

*According to the above update rule, each node i can distributedly implement (86) by performing the following*

$$x_i^{r+1} = x_i^r - \frac{1}{2\sum_{j:j\sim i} \sigma_{ij}^2 + \beta_i^2} \left(\frac{1}{M}(\nabla f_i(x_i^r) - \nabla f_i(x_i^{r-1}))\right. \tag{87}$$
$$\left. - 2\sum_{j:j\sim i} \sigma_{ij}^2 x_j^r - \beta_i^2 x_i^r + \beta_i^2 x_i^{r-1} + \sum_{j:j\sim i} \sigma_{ij}^2(x_j^{r-1} + x_i^{r-1})\right).$$

*It is also easy to see that the Chebyshev iteration in xFILTER can be implemented distributedly, since the R matrix defined in (74) preserves the network structure. To see how we can compute the $d^r$ vector distributedly, we first note that $d^{-1} = -\Upsilon^{-2}\nabla f(0)$. Then suppose we know $d^{r-1}$, by combining (79) and (78) we have*

$$d^r = d^{r-1} + (x^r - x^{r-1}) - \Upsilon^{-2}(\nabla f(x^r) - \nabla f(x^{r-1})) - \Upsilon^{-2} F^T \Sigma^2 F x^r.$$

*Therefore each $d_i^r$ can be updated as*

$$d_i^r = d_i^{r-1} + (x_i^r - x_i^{r-1}) - \frac{1}{M\beta_i^2}(\nabla f_i(x_i^r) - \nabla f_i(x_i^{r-1})) + \sum_{j:j\sim i} \frac{\sigma_{ij}^2}{\beta_i^2}(x_i^r - x_j^r). \tag{88}$$

*Combining the above expression with the expression in (75) for computing $Rd^r$, it is clear that all the computation only involves in local communication and local gradient computation.*

*These observations also suggest that for a general choice of parameter matrix $\Sigma^2 \succ 0$, both D-GPDA and xFILTER are in class $\mathcal{A}$. Further, if $\Sigma^2$ is a multiple of identity matrix (i.e., there exists $\sigma^2 > 0$ such that $\Sigma^2 = \sigma^2 I_E$), then the computations in (87) and (88) only involve the sum of neighboring iterates, therefore both algorithms belong to class $\mathcal{A}'$ as well.*

# 5 The Convergence Rate Analysis

In this section we provide the analysis steps of the convergence rate of the D-GPDA and xFILTER. All the proofs of the results can be found in the appendix. Note that we use the primal-dual representation discussed in Section 4.3 for xFILTER, so that it can be analyzed together with the D-GPDA.

**Step 1.** We first analyze the dynamics of the dual variable.

**Lemma 5.1** *Suppose that $f(x)$ is in class $\mathcal{P}_L^M$. Then for all $r \geq 0$, the iterates of D-GPDA satisfy*

$$\|\lambda^{r+1} - \lambda^r\|_{\Sigma^{-2}}^2 \leq 2\kappa \left( \frac{\|\Upsilon^{-1}L(x^r - x^{r-1})\|^2}{M^2} + \|w^{r+1}\|_H^2 \right). \tag{89}$$

*Further, for all $r \geq 0$, the iterates of xFILTER satisfy*

$$\|\lambda^{r+1} - \lambda^r\|_{\Sigma^{-2}}^2 \leq \widetilde{\kappa} \left( \frac{3}{M^2} \|\Upsilon^{-1}L(x^r - x^{r-1})\|^2 + 3\|w^{r+1}\|_{\Upsilon^2}^2 + 3\|\Upsilon R(\epsilon^{r+1} - \epsilon^r)\|^2 \right). \tag{90}$$

*In the above we have defined the following*

$$\kappa := \frac{1}{\lambda_{\min}(\Sigma F H^{-1} F^T \Sigma)}, \quad \widetilde{\kappa} := \frac{1}{\lambda_{\min}(\Sigma F \Upsilon^{-2} F^T \Sigma)} = \frac{1}{\lambda_{\min}(\mathcal{L}_G)} \tag{91a}$$

$$w^{r+1} := (x^{r+1} - x^r) - (x^r - x^{r-1}). \tag{91b}$$

**Step 2.** In this step we analyze the dynamics of the AL.

**Lemma 5.2** *Suppose that $f(x)$ is in class $\mathcal{P}_L^M$. Then for all $r \geq 0$, the iterates of D-GPDA satisfy*

$$\mathsf{AL}^{r+1} - \mathsf{AL}^r \leq -\frac{1}{2}\|x^{r+1} - x^r\|_{\Delta + 2\Upsilon^2 - L/M}^2$$
$$+ \kappa \left( \frac{2}{M^2} \|\Upsilon^{-1}L(x^r - x^{r-1})\|^2 + 2\|w^{r+1}\|_H^2 \right). \tag{92}$$

*Further, for all $r \geq 0$, the iterates of xFILTER satisfy*

$$\mathsf{AL}^{r+1} - \mathsf{AL}^r \leq -\frac{1}{2}\|x^{r+1} - x^r\|_{\Upsilon^2 R - \frac{L}{M}}^2 + \langle \Upsilon^2 R \epsilon^{r+1}, x^{r+1} - x^r \rangle \tag{93}$$
$$+ \widetilde{\kappa} \left( \frac{3}{M^2} \|\Upsilon^{-1}L(x^r - x^{r-1})\|^2 + 3\|w^{r+1}\|_{\Upsilon^2}^2 + 3\|\Upsilon R(\epsilon^{r+1} - \epsilon^r)\|^2 \right).$$

Before moving forward, we provide bounds for the important parameters $\kappa$ and $\tilde{\kappa}$. From (91a) we can express $\kappa$ as

$$\kappa = \frac{1}{\lambda_{\min}(\Sigma F \Upsilon^{-1} \left(\Upsilon^{-1}B^T \Sigma^2 B \Upsilon^{-1} + I\right)^{-1} \Upsilon^{-1} F^T \Sigma)}$$
$$= \frac{1}{\lambda_{\min}((\Upsilon^{-1}B^T \Sigma^2 B \Upsilon^{-1} + I)^{-1} \mathcal{L}_G)}$$
$$\overset{(23)}{=} \frac{1}{\lambda_{\min}\left((-\mathcal{L}_G + 2\Upsilon^{-1}\Delta\Upsilon^{-1} + I)^{-1}\mathcal{L}_G\right)}. \tag{94}$$

Similar derivation applies for $\tilde{\kappa}$. In summary we have

$$\kappa \leq \frac{\lambda_{\max}(2\Upsilon^{-1}\Delta\Upsilon^{-1}+I)}{\underline{\lambda}_{\min}(\mathcal{L}_G)}, \quad \widetilde{\kappa} = \frac{1}{\underline{\lambda}_{\min}(\mathcal{L}_G)}. \tag{95}$$

**Step 3.** In this step, we analyze the error sequences $\{\epsilon^{r+1}\}$ generated by the xFILTER. First we have the following well-known result on the behavior of the Chebyshev iteration; see, e.g., [54, Chapter 6] and [55, Theorem 1, Chapter 7].

**Lemma 5.3** *Consider using the Chebyshev iteration* (77) *to solve* $Rx = d^r$. *Define* $x^{r+1}_* = R^{-1}d^r$, *with*

$$R := \Upsilon^{-2}(F^T\Sigma^2F + \Upsilon^2). \tag{96}$$

*Define the following constants:*

$$\xi(R) := \frac{\lambda_{\min}(R)}{\lambda_{\max}(R)} \leq 1,\ \xi(\Upsilon^2) := \frac{\lambda_{\min}(\Upsilon^2)}{\lambda_{\max}(\Upsilon^2)} \leq 1,\ \theta(R) := \lambda_{\min}(R) + \lambda_{\max}(R). \tag{97}$$

*Choose the following parameters:*

$$\tau = \frac{2}{\theta(R)},\ \alpha_1 = 2,\ \alpha_{t+1} = \frac{4}{4-\rho_0^2\alpha_t},\ \rho_0 = \frac{1-\xi(R)}{1+\xi(R)}.$$

*Then for any* $\eta \in (0,1)$, *in order to achieve the following accuracy*

$$\|u_Q - x^{r+1}_*\|^2_{\Upsilon^2} \leq \eta\|u_0 - x^{r+1}_*\|^2_{\Upsilon^2}, \tag{98}$$

*it requires the following number of iterations*

$$Q \geq -\frac{1}{4}\ln(\eta/4)\sqrt{1/\xi(R)}.$$

Recall that in Algorithm 2 the initial and final solutions for the Chebyshev iteration are assigned to $x^r$ and $x^{r+1}$, respectively. Define $\widetilde{\epsilon}^r := u_0 - x^{r+1}_*$, we have

$$Rx^r = Ru_0 = R(u_0 - x^{r+1}_*) + Rx^{r+1}_* := R\widetilde{\epsilon}^r + d^r, \forall\ r \geq -1.$$

Plugging in the definition of $d^r$ in (81), we obtain

$$R\widetilde{\epsilon}^r = Rx^r + \Upsilon^{-2}(\nabla f(x^r) + F^T\lambda^r - \Upsilon^2 x^r). \tag{99}$$

Using the definition of $\epsilon^{r+1}$ in (82b), and the fact that $R$ is invertible, we obtain the following key relationship

$$\epsilon^{r+1} - \widetilde{\epsilon}^r = x^{r+1} - x^r, \forall\ r \geq -1. \tag{100}$$

Recall that $\epsilon^{r+1} := x^{r+1} - x^{r+1}_*$, and $x^{r+1} = u_Q$, $x^r = u_0$, then (98) implies

$$\|\epsilon^{r+1}\|^2_{\Upsilon^2} \leq \eta\|\widetilde{\epsilon}^r\|^2_{\Upsilon^2}. \tag{101}$$

By combining Lemma 5.3, (100) and (101), the following result provides some essential relationships between the error sequences $\{\epsilon^{r+1}\}$ incurred by running finite number of C-iterations, with the outer-loop iterations $\{x^{r+1}\}$.

**Lemma 5.4** *Choose the inner iteration of xFILTER as*

$$Q = -\frac{1}{4}\ln\left(\frac{\theta^2}{16 + 128M\max\{\lambda_{\max}(\Upsilon^2 R), 1\}}\right)\sqrt{1/\xi(R)}. \tag{102}$$

*where $\theta := \xi(\Upsilon^2 R)\xi(\Upsilon^2) \times \min\{1, \lambda_{\min}(\Upsilon^2)\}$. Then we have the following inequalities*

$$\|\Upsilon^2 R\epsilon^{r+1}\|^2 \leq \frac{1}{16M}\|x^{r+1} - x^r\|^2_{\Upsilon^2 R}, \tag{103a}$$

$$\|\epsilon^{r+1}\|^2_{\Upsilon^2 R} \leq \frac{1}{16M}\|x^{r+1} - x^r\|^2_{\Upsilon^2 R}, \tag{103b}$$

$$\|\Upsilon R\epsilon^{r+1}\|^2 \leq \frac{1}{16M}\|x^{r+1} - x^r\|^2_{\Upsilon^2 R}, \tag{103c}$$

$$\langle \Upsilon^2 R\epsilon^{r+1}, x^{r+1} - x^r\rangle \leq \frac{3}{16}\|x^{r+1} - x^r\|^2_{\Upsilon^2 R}, \tag{103d}$$

$$\langle \Upsilon^2 R\epsilon^r, x^{r+1} - x^r\rangle \leq \frac{1}{8}\|x^r - x^{r-1}\|^2_{\Upsilon^2 R} + \frac{1}{16}\|x^{r+1} - x^r\|^2_{\Upsilon^2 R}. \tag{103e}$$

## 5.1   Proof of Lemma 5.4

**Proof.** Let us choose

$$\eta = \theta^2/(4 + 32M\max\{\lambda_{\max}(\Upsilon^2 R), 1\}). \tag{104}$$

Then from Lemma 5.3, it is clear that if $Q$ satisfies (102), then

$$\|\epsilon^{r+1}\|^2_{\Upsilon^2} \leq \eta\|\tilde{\epsilon}^r\|^2_{\Upsilon^2}. \tag{105}$$

Note that $\Upsilon^2 R = F^\top \Sigma^2 F + \Upsilon^2 \succ 0$, then it follows that

$$\|\Upsilon^2 R\epsilon^{r+1}\|^2 \leq \frac{\lambda_{\max}(R\Upsilon^2\Upsilon^2 R)}{\lambda_{\min}(\Upsilon^2)}\|\epsilon^{r+1}\|^2_{\Upsilon^2}$$

$$\overset{(101)}{\leq} \frac{\eta\lambda_{\max}(R\Upsilon^2\Upsilon^2 R)}{\lambda_{\min}(\Upsilon^2)}\|\tilde{\epsilon}^r\|^2_{\Upsilon^2} \leq \frac{\eta\lambda_{\max}(R\Upsilon^2\Upsilon^2 R)\lambda_{\max}(\Upsilon^2)}{\lambda_{\min}(\Upsilon^2)}\|\tilde{\epsilon}^r\|^2$$

$$\leq \frac{\eta\lambda_{\max}(R\Upsilon^2\Upsilon^2 R)\lambda_{\max}(\Upsilon^2)}{\lambda_{\min}(R\Upsilon^2\Upsilon^2 R)\lambda_{\min}(\Upsilon^2)}\|\Upsilon^2 R\tilde{\epsilon}^r\|^2 \leq \eta\theta^{-2}\|\Upsilon^2 R\tilde{\epsilon}^r\|^2.$$

Using the above relation, we can then obtain the following

$$\|\Upsilon^2 R\epsilon^{r+1}\|^2 \leq 2\eta\theta^{-2}(\|\Upsilon^2 R\epsilon^{r+1}\|^2 + \|\Upsilon^2 R(\epsilon^{r+1} - \tilde{\epsilon}^r)\|^2)$$

$$\overset{(100)}{\leq} 2\eta\theta^{-2}(\|\Upsilon^2 R\epsilon^{r+1}\|^2 + \|\Upsilon^2 R(x^{r+1} - x^r)\|^2).$$

Therefore, we obtain

$$\|\Upsilon^2 R\epsilon^{r+1}\|^2 \leq 2\eta\theta^{-2}/(1 - 2\eta\theta^{-2})\|\Upsilon^2 R(x^{r+1} - x^r)\|^2.$$

Plugging the definition of $\eta$ in (104), we have

$$\|\Upsilon^2 R\epsilon^{r+1}\|^2 \le \lambda_{\max}(\Upsilon^2 R)2\eta\theta^{-2}/(1-2\eta\theta^{-2})\|x^{r+1}-x^r\|^2_{\Upsilon^2 R}$$

$$\overset{(104)}{\le} 1/(16M)\|x^{r+1}-x^r\|^2_{\Upsilon^2 R}, \quad \forall\, r \ge -1.$$

To obtain the second inequality, notice that

$$\|\epsilon^{r+1}\|^2_{\Upsilon^2 R} \le \theta^{-1}\eta\|\widetilde{\epsilon}^r\|^2_{\Upsilon^2 R} \le \theta^{-2}\eta\|\widetilde{\epsilon}^r\|^2_{\Upsilon^2 R} \tag{106}$$

where the last inequality is due to the fact that $\theta \le 1$. Then repeating the above derivation we can obtain the desired result. The third inequality in (103) can be derived in a similar way, and the last two in (103) can be obtained by using Cauchy-Swartz inequality. **Q.E.D.**

Clearly, using the Chebyshev iteration is one critical step that ensures fast reduction of the error $\{\epsilon^{r+1}\}$. In particular, to achieve constant reduction of error, the total number of required Chebyshev iteration is proportional to $\sqrt{1/\xi(R)}$, rather than $1/\xi(R)$ in conventional iterative scheme such as the Richardson's iteration [54]. Such a choice enables the final bound to be dependent on $\sqrt{1/\xi(\mathcal{G})}$, rather than $1/\xi(\mathcal{G})$.

**Step 4.** Let us construct the following potential functions (parameterized by constants $c, \widetilde{c} > 0$)

$$P_c(x^{r+1}, x^r, \lambda^{r+1}) := \mathsf{AL}^{r+1} + \frac{2\kappa}{M^2}\|\Upsilon^{-1}L(x^{r+1}-x^r)\|^2$$
$$+ \frac{c}{2}\left(\|\Sigma F x^{r+1}\|^2 + \|x^{r+1}-x^r\|^2_{H+L/M}\right). \tag{107a}$$

$$\widetilde{P}_{\widetilde{c}}(x^{r+1}, x^r, \lambda^{r+1}) := \mathsf{AL}^{r+1} + \frac{3\widetilde{\kappa}}{M^2}\|\Upsilon^{-1}L(x^{r+1}-x^r)\|^2 \tag{107b}$$
$$+ \frac{3\widetilde{\kappa}}{8}\|x^{r+1}-x^r\|^2_{\Upsilon^2 R} + \frac{\widetilde{c}}{2}\left(\|\Sigma F x^{r+1}\|^2 + \|x^{r+1}-x^r\|^2_{\Upsilon^2+\frac{\Upsilon^2 R}{4}+\frac{L}{M}}\right).$$

For notational simplicity we will denote them as $P^{r+1}$ and $\widetilde{P}^{r+1}$, respectively. In the following we show that when the algorithm parameters are chosen properly, the potential functions will decrease along the iterations.

**Lemma 5.5** *Suppose that $f(x)$ is in class $\mathcal{P}_L^M$, and that the parameters of D-GPDA are chosen as below*

$$c = \max\{6\kappa, 1\}, \quad \Upsilon^2 \succeq \frac{L\Upsilon^{-2}L}{M^2}, \tag{108a}$$

$$\frac{1}{2}\left(\Delta + \Upsilon^2\right) - \frac{L}{M} - \frac{4\kappa}{M^2}L\Upsilon^{-2}L - \frac{2cL}{M} \succeq 0. \tag{108b}$$

*Then for all $r \ge 0$, we have*

$$P^r - P^{r+1} \ge \frac{1}{4}\|x^{r+1}-x^r\|^2_{\Delta+\Upsilon^2} + \kappa\|w^{r+1}\|^2_H. \tag{109}$$

**Lemma 5.6** *Suppose that $f(x)$ is in class $\mathcal{P}_L^M$, $Q$ is chosen according to (102), and the rest of the parameters of xFILTER are chosen as below*

33

$$\tilde{c} = 8\tilde{\kappa} = \frac{8}{\underline{\lambda}_{\min}(\Sigma F \Upsilon^{-2} F^T \Sigma)}, \quad \Upsilon^2 \succeq \frac{L\Upsilon^{-2}L}{M^2}, \tag{110a}$$

$$(1/4 - 3\tilde{\kappa} - \tilde{c})\Upsilon^2 R - (1 + 2\tilde{c})L/M - \frac{6\tilde{\kappa}}{M^2}L\Upsilon^{-2}L \succeq 0. \tag{110b}$$

*Then for all $r \geq 0$, we have*

$$\widetilde{P}^r - \widetilde{P}^{r+1} \geq \frac{1}{8}\|x^{r+1} - x^r\|_{\Upsilon^2 R}^2 + \tilde{\kappa}\|w^{r+1}\|_{\Upsilon^2}^2. \tag{111}$$

**Step 5.** Next we show the lower and upper boundedness of the potential function.

**Lemma 5.7** *Suppose that $f(x)$ is in class $\mathcal{P}_L^M$ and the parameters are chosen according to (108). Then the iterates generated by D-GPDA satisfy*

$$P^{r+1} \geq \underline{f} > -\infty, \quad \forall \, r > 0, \tag{112a}$$

$$P^0 \leq f(x^0) + \frac{2}{M}d_0^T L^{-1}d_0, \tag{112b}$$

*where $d_0$ is defined in (48).*
  *Similarly, for xFILTER the function $\widetilde{P}^{r+1}$ has the same expression as in (112a), and*

$$\widetilde{P}^0 \leq f(x^0) + \frac{5}{M}d_0^T L^{-1}d_0. \tag{113}$$

**Step 6.** We are ready to derive the final bounds for the convergence rate of the proposed algorithms.

**Theorem 5.1** *Suppose that $f(x)$ is in class $\mathcal{P}_L^M$ and the parameters are chosen according to (108). Let $T$ denote an iteration index in which D-GPDA satisfies*

$$e(T) := \min_{r \in [T]} \left\| 1/M \sum_{i=1}^M \nabla f_i(x_i^r) \right\|^2 + \|\Sigma F x^r\|^2 \leq \epsilon. \tag{114}$$

*Then we have the following bound for the error:*

$$\epsilon \leq C_1 \times \frac{C_2}{T}, \quad with \ C_1 := 8\left(f(x^0) - \underline{f} + \frac{2}{M}d_0^T L^{-1}d_0\right)$$

$$C_2 := 4 \sum_{(i,j):i\sim j} \sigma_{ij}^2 + \sum_{i=1}^M \beta_i^2 + 4. \tag{115}$$

*Similarly, for xFILTER when the parameters are chosen according to (110) and (102), the same equation*

$$\epsilon \leq \widetilde{C}_1 \times \frac{\widetilde{C}_2}{T_r} \tag{116}$$

*holds true (with $T_r$ denoting the total number of outer iterations), with the following constants*

$$\widetilde{C}_1 := f(x^0) - \underline{f} + \frac{5}{M} d_0^T L^{-1} d_0 \tag{117a}$$

$$\widetilde{C}_2 := 128 \left( \sum_{i=1}^{M} \beta_i^2 + 3 + \frac{1}{32\widetilde{\kappa}} \right). \tag{117b}$$

We note that one key difference between the two rates is that, the constant $C_2$ for D-GPDA depends explicitly on $\sigma_e$'s, while its counterpart for xFILTER depends on $1/\widetilde{\kappa}$ instead. Further, for xFILTER, the constant $\widetilde{\kappa}$ in (95) only depends on $\underline{\lambda}_{\min}(\mathcal{L}_G)$, while for D-GPDA $\kappa$ is further dependent on $\lambda_{\max}(\Upsilon^{-1}\Delta\Upsilon^{-1})$. These properties will be leveraged later when choosing algorithm parameters to ensure that optimal rates for different problems and networks are obtained.

# 6 Rate Bounds and Tightness

In this section we provide explicit choices of various parameters, and discuss the tightness of the resulting bounds for D-GPDA and xFILTER.

## 6.1 Parameter Selection and Rate Bounds for D-GPDA

Let us pick the following parameters for D-GPDA

$$\sigma_{ij}^2 = \frac{\beta^2 \sqrt{L_i L_j}}{\sqrt{d_i d_j}}, \ \ \Upsilon^2 = \beta^2 L, \ \ \beta^2 = \frac{80 \max\{\lambda_{\max}(W), 1\}}{\min\{\underline{\lambda}_{\min}(\mathcal{L}_G), 1\}M}. \tag{118}$$

It follows that the following relations hold

$$\Delta = \beta^2 W, \quad \beta_i^2 = \beta^2 L_i, \ \forall \ i, \quad \kappa \overset{(95)}{\leq} \frac{1 + 2\lambda_{\max}(W)}{\min\{\underline{\lambda}_{\min}(\widetilde{L}), 1\}}. \tag{119}$$

In the above definitions, we have defined $W \in \mathbb{R}^M$ as a diagonal matrix with

$$[W]_{ii} = \frac{\sqrt{L_i}}{\sqrt{d_i}} \sum_{q:q\sim i} \frac{\sqrt{L_q}}{\sqrt{d_q}},$$

and that

$$[\mathcal{L}_G]_{ij} = \begin{cases} \sum_{q:q\sim i} \frac{1}{\sqrt{d_q d_i}} & \text{if } i = j \\ -\frac{1}{\sqrt{d_i d_j}} & \text{if } (ij) \in \mathcal{E}, i \neq j \\ 0 & \text{otherwise.} \end{cases} \tag{120}$$

Note that when $d_i = d_j$, $\forall \ i, j$, we have $\mathcal{L}_G = \mathcal{L}$. We have the following result.

**Theorem 6.1** *Consider using D-GPDA to solve problems in class $(\mathcal{P}_L^M, \mathcal{N}_D^M)$, using parameters in (118). Then the condition (108b) will be satisfied. Further, to achieve $e(T) \leq \epsilon$, it requires at most the following*

*number of iterations*

$$T \le \frac{8}{\epsilon}\left(f(x^0) - \underline{f} + \frac{2}{M}\|d_0\|^2_{L^{-1}}\right) \times C_2 \tag{121}$$

*where $C_2$ is given by [with $W$ and $\widetilde{\mathcal{L}}$ defined in (120)]*

$$C_2 \le \frac{320\max\{\lambda_{\max}(W),1\}}{\min\{\underline{\lambda}_{\min}(\mathcal{L}_G),1\}} \sum_{(i,j):i\sim j}\left(\frac{\sqrt{L_i L_j}}{\sqrt{d_i d_j}M} + \frac{\bar{L}}{4}\right) + 4. \tag{122}$$

**Proof.** For D-GPDA, use the parameters in (118), we have

$$c \le \frac{6 + 12\lambda_{\max}(W)}{\min\{\underline{\lambda}_{\min}(\mathcal{L}_G),1\}}, \quad \Upsilon^2 = \frac{80\max\{\lambda_{\max}(W),1\}}{\min\{\underline{\lambda}_{\min}(\mathcal{L}_G),1\}M}L.$$

Therefore to ensure condition (108b), it suffices to ensure the following

$$\frac{40\max\{\lambda_{\max}(W),1\}}{\min\{\underline{\lambda}_{\min}(\mathcal{L}_G),1\}M}L - \frac{(4 + 8\lambda_{\max}(W))}{M80\max\{\lambda_{\max}(W),1\}}L - \frac{1}{M}L - \frac{6 + 12\lambda_{\max}(W)}{\min\{\underline{\lambda}_{\min}(\mathcal{L}_G),1\}}\frac{2}{M}L \succ 0. \tag{123}$$

It is easy to check that this inequality will be satisfied using the above choice of parameters. Using these choices, we can obtain the desired expression for $C_2$. **Q.E.D.**

## 6.2  Parameter Selection and Rate Bounds for xFILTER

First, recall that we have defined the matrix $\widetilde{L}$ and $\widehat{\mathcal{L}}$ as follows [see the definition in (25)]

$$\widetilde{\mathcal{L}} = L^{-1/2}P^{-1/2}F^T KFP^{-1/2}L^{-1/2},$$
$$\widehat{\mathcal{L}} = L^{-1/2}F^T KFL^{-1/2}.$$

Below we will provide two different choices of parameters.
**Choice I.** We will focus on a class of graphs such that there exists an *absolute* constant $k > 0$ such that the following holds (i.e., the degrees of the nodes are not quite different from their averages):

$$kP \succeq \bar{d}I_M. \tag{124}$$

The above condition says that the degrees of the nodes are not quite different from their averages. For example the following graphs satisfy (124): Complete graph ($k = 1$), star graph ($k = 2$), grid graph ($k = 2$), cubic graph ($k = 1$), path graph ($k = 2$), and any regular graph ($k = 1$).

For the class of graphs satisfy (124), let us pick the parameters for xFILTER as follows

$$\Sigma^2 = \frac{48 \times 96k}{\sum_i d_i \underline{\lambda}_{\min}(\widetilde{\mathcal{L}})}K, \quad \Upsilon^2 = \frac{96k}{\sum_i d_i}P^{1/2}LP^{1/2}. \tag{125}$$

Using the above choice, we have

$$\beta_i^2 = \frac{96 L_i d_i k}{\sum_i d_i} \tag{126}$$

and that the matrix $\Upsilon$ satisfies the following

$$\Upsilon^2 = \frac{96k}{\sum_i d_i} P^{1/2} L P^{1/2} \succeq \frac{96}{M} L. \tag{127}$$

Plugging these choices to the generalized Laplacian $\mathcal{L}_G$ in (23) we obtain

$$\mathcal{L}_G = \Upsilon^{-1} F^T \Sigma^2 F \Upsilon^{-1}$$
$$= \frac{48}{\underline{\lambda}_{\min}(\widetilde{\mathcal{L}})} L^{-1/2} P^{-1/2} F^T K F P^{-1/2} L^{-1/2} = \frac{48}{\underline{\lambda}_{\min}(\widetilde{\mathcal{L}})} \widetilde{\mathcal{L}}. \tag{128}$$

Therefore by (95) we have

$$\tilde{\kappa} = \frac{\lambda_{\min}(\widetilde{\mathcal{L}})}{48\underline{\lambda}_{\min}(\widetilde{\mathcal{L}})} = \frac{1}{48}. \tag{129}$$

Also in this case we have

$$R = \Upsilon^{-2} F^T \Sigma^2 F + I = \frac{48}{\underline{\lambda}_{\min}(\widetilde{\mathcal{L}})} P^{-1/2} L^{-1} P^{-1/2} F^T K F + I.$$

By noting that the matrix $P^{-1/2} L^{-1} P^{-1/2} F^T K F$ and $\widetilde{\mathcal{L}}$ has the same set of eigenvalues, we obtain

$$\lambda_{\max}(R) \leq \left( \frac{48\lambda_{\max}(\widetilde{\mathcal{L}})}{\underline{\lambda}_{\min}(\widetilde{\mathcal{L}})} + 1 \right) \leq \frac{50}{\xi(\widetilde{\mathcal{L}})}, \ \lambda_{\min}(R) = 1, \tag{130a}$$

$$\xi(R) \geq 1 / \left( \frac{48\lambda_{\max}(\widetilde{\mathcal{L}})}{\underline{\lambda}_{\min}(\widetilde{\mathcal{L}})} + 1 \right) \geq \frac{\xi(\widetilde{\mathcal{L}})}{50}. \tag{130b}$$

**Choice II.** For general graphs not necessarily satisfying (124), let us pick the parameters for xFILTER as follows

$$\Sigma^2 = \frac{48 \times 96}{M \underline{\lambda}_{\min}(\widehat{\mathcal{L}})} K, \quad \Upsilon^2 = \frac{96}{M} L. \tag{131}$$

Using the above choice, we have

$$\beta_i^2 = \frac{96 L_i}{M}. \tag{132}$$

We have that

$$\mathcal{L}_G = \Upsilon^{-1} F^T \Sigma^2 F \Upsilon^{-1} = \frac{48}{\underline{\lambda}_{\min}(\widehat{\mathcal{L}})} L^{-1/2} F^T K F L^{-1/2} = \frac{48}{\underline{\lambda}_{\min}(\widehat{\mathcal{L}})} \widehat{\mathcal{L}}. \tag{133}$$

Therefore by (95) we have

$$\tilde{\kappa} = \frac{\lambda_{\min}(\widehat{\mathcal{L}})}{48\underline{\lambda}_{\min}(\widehat{\mathcal{L}})} = \frac{1}{48}. \tag{134}$$

Also in this case we have

$$R = \Upsilon^{-2} F^T \Sigma^2 F + I = \frac{48}{\underline{\lambda}_{\min}(\widehat{\mathcal{L}})} L^{-1} F^T K F + I.$$

By noting that the matrix $L^{-1}F^T K F$ and $\widehat{\mathcal{L}}$ has the same set of eigenvalues, we obtain

$$\lambda_{\max}(R) \leq \left( \frac{48\lambda_{\max}(\widehat{\mathcal{L}})}{\underline{\lambda}_{\min}(\widehat{\mathcal{L}})} + 1 \right) \leq \frac{50}{\xi(\widehat{\mathcal{L}})}, \ \lambda_{\min}(R) = 1, \tag{135a}$$

$$\xi(R) \geq 1 / \left( \frac{48\lambda_{\max}(\widehat{\mathcal{L}})}{\underline{\lambda}_{\min}(\widehat{\mathcal{L}})} + 1 \right) \geq \frac{\xi(\widehat{\mathcal{L}})}{50}. \tag{135b}$$

**Remark 6.1 (Choices of Parameters)** *The main difference between the above two choices of parameters is whether $\Upsilon^2$ is scaled with the degree matrix or not. The resulting bounds are also dependent on the spectral gap for $\widetilde{L}$ and $\widehat{\mathcal{L}}$, one inversely scaled with the degree matrix, and the other does not. Note that the spectral gap of $\widetilde{L}$ and $\widehat{\mathcal{L}}$ may not be the same. For example for a star graph with $L_i = L_j$, $\xi(\widehat{\mathcal{L}}) = \mathcal{O}(1/M)$ but $\xi(\widehat{\mathcal{L}}) = \mathcal{O}(1)$. Therefore one has to be careful in choosing these parameters so that $\xi(R)$ is made as large as possible.*

*Additionally, since we are mainly interested in choosing the optimal parameters so that the resulting rate bounds will be optimal in their dependency on problem parameters, the absolute constants in the above parameter choices have not been optimized.*

The following result is a direct consequence of the second part of Theorem 5.1.

**Theorem 6.2** *Consider using xFILTER to solve problems in class $(\mathcal{P}_L^M, \mathcal{N}_D^M)$, then the following holds.*
**Case I.** *Further restricting $\mathcal{N}_D^M$ to a subclass satisfying (124). If parameters in (125) is used, then the condition (110b) will be satisfied. Further, to achieve $e(T) \leq \epsilon$, it requires at most the following number of iterations (where $T$ denotes the total iterations of the xFILTER algorithm)*

$$
\begin{aligned}
T &\leq \frac{1}{\epsilon} \left( f(x^0) - \underline{f} + \frac{5}{M} \|d_0\|_{L^{-1}}^2 \right) \times \widetilde{C}_2 \\
&\quad \times \frac{1}{4} \ln \left( \frac{16 + 128M \max\{\lambda_{\max}(\Upsilon^2 R), 1\}}{\theta^2} \right) \sqrt{1/\xi(R)} \\
&\leq \frac{1}{\epsilon} \left( f(x^0) - \underline{f} + \frac{5}{M} \|d_0\|_{L^{-1}}^2 \right) \times \widetilde{C}_2 \\
&\quad \times \frac{1}{4} \ln \left( \frac{50^2 (ML_{\max}/L_{\min})^4 \times (16 + 128M \max\{50 \times 96k L_{\max}, 1\})}{\xi^3(\widetilde{\mathcal{L}}) \times \min\{1, 96^2 k^2 L_{\min}^2/M^2\}} \right) \sqrt{50/\xi(\widetilde{\mathcal{L}})} \tag{136}
\end{aligned}
$$

where $\widetilde{C}_2$ is given by

$$\widetilde{C}_2 \le 128 \left( \frac{96k}{\sum_{i=1}^{M} d_i} \sum_{i=1}^{M} d_i L_i + 19 \right). \tag{137}$$

**Case II.** *Suppose parameters in* (131) *are used. Then the condition* (110b) *will be satisfied. Further, to achieve* $e(T) \le \epsilon$, *it requires at most the following number of iterations*

$$T \le \frac{1}{\epsilon} \left( f(x^0) - \underline{f} + \frac{5}{M} \|d_0\|_{L^{-1}}^2 \right) \times \widetilde{C}_2$$

$$\times \frac{1}{4} \ln \left( \frac{50^2 (L_{\max}/L_{\min})^4 \times (16 + 128 M \max\{50 \times 96 L_{\max}/M, 1\})}{\xi^3(\widetilde{\mathcal{L}}) \times \min\{1, 96^2 L_{\min}^2/M^2\}} \right) \sqrt{50/\xi(\widetilde{\mathcal{L}})} \tag{138}$$

*where* $\widetilde{C}_2$ *is given by*

$$\widetilde{C}_2 \le 128 \left( \frac{96}{M} \sum_{i=1}^{M} L_i + 19 \right). \tag{139}$$

We note that compared with the results in Theorem 5.1, the additional multiplicative term in (136) accounts for the Chebyshev iterations that are needed for every iteration $t$. It is interesting to observe that comparing with the previous result, the constant $\widetilde{C}_2$ in (137) is independent on any graph parameters. Such a desirable property turns out to be crucial for obtaining tight rate bounds.

## 6.3 Tightness of the Upper Rate Bounds

In this section, we present some tightness results of the upper rate bounds for our proposed D-GPDA and xFILTER. In particular, we compare the expressions derived in Theorem 6.1 – 6.2, and the lower bounds derived in Section 3, over different kinds of graphs and for different problems. We will mainly focus on the case with uniform Lipschitz constants, i.e., $L_i = U, \forall i$. WE will briefly discuss the case of non-uniform Lipschitz constants at the end of this section.

First, we consider the problem class $\mathcal{P}_U^M$ with the following properties:

$$L_1 = L_2 = \cdots L_M = \frac{1}{M} \sum_{i=1}^{M} L_i := U, \quad L = U I_M. \tag{140}$$

It follows that in this case $\widetilde{\mathcal{L}} = \mathcal{L}$, and $\widehat{\mathcal{L}} = P^{1/2} \mathcal{L} P^{1/2}$. Let us first make some useful observations.

**Remark 6.2** *Let us specialize the parameter choices for D-GPDA algorithm in* (118) *and derive the bounds for* $C_2$ *in* (122) *for two special graphs.*
**Complete graph.** *For complete graphs we have* $d_i = d_j = M - 1, \forall i, j$, *which implies that* $\mathcal{L}_G = \mathcal{L}$, *so* $\underline{\lambda}_{\min}(\mathcal{L}_G) = M/(M-1)$. *Because* $L_i = L_i = U, \forall i, j$, *we have* $W = I_M$. *Therefore using the expression* (122) *we obtain the following:*

$$C_2^{\text{comp}} \le 400U + 4. \tag{141}$$

**Cycle graph.** *For cycle graph we have $d_i = d_j = 2, \forall\ i, j$, which implies that $\mathcal{L}_G = \mathcal{L}$, and $\underline{\lambda}_{\min}(\mathcal{L}_G) \geq 1/M^2$. Because $L_i = L_i = U, \forall\ i, j$, we have $W = I_M$. Therefore using the expression (122) we obtain the following:*

$$C_2^{\text{cycle}} \leq 240UM^2 + 4. \tag{142}$$

*It is clear that for cycle graph whose diameter is in $\mathcal{O}(M)$, the rate bounds is very large.*

**Remark 6.3** *Let us specialize the parameter choices for xFILTER algorithm in (125) and derive the bounds for $\widetilde{C}_2 \times 1/\sqrt{\xi(\widetilde{\mathcal{L}})}$ in (137) for the following special graphs. Note that because uniform $L_i$'s are assumed, we have $\widetilde{\mathcal{L}} = \mathcal{L}$.*

**Complete graph.** *Complete graphs satisfy (124) with $k = 1$. It also satisfies $\underline{\lambda}_{\min}(\widetilde{\mathcal{L}}) = M/(M-1) \geq 1$. Therefore using the expression (137) we obtain the following:*

$$\widetilde{C}_2^{\text{comp}} \times \frac{1}{\sqrt{\xi(\widetilde{\mathcal{L}})}} \leq 12500U + 2560. \tag{143}$$

**Grid graph.** *Grid graphs satisfy (124) with $k = 2$. It also satisfies $\underline{\lambda}_{\min}(\widetilde{\mathcal{L}}) \geq 1/M$. Therefore using the expression (137) we obtain the following:*

$$\widetilde{C}_2^{\text{grid}} \times \frac{1}{\sqrt{\xi(\widetilde{\mathcal{L}})}} \leq (12500U + 2560) \times \sqrt{M}. \tag{144}$$

**Star graph.** *Star graphs satisfy (124) with $k = 2$. It also has $\xi(\widetilde{\mathcal{L}}) = 1/2$. Therefore using the expression (137) we obtain the following:*

$$\widetilde{C}_2^{\text{star}} \times \frac{1}{\sqrt{\xi(\widetilde{\mathcal{L}})}} \leq (12500U + 2560) \times \sqrt{2}. \tag{145}$$

**Geometric graph.** *For geometric graphs which place the nodes uniformly in $[0, 1]^2$ and connect any two nodes separated by a distance less than a radius $R \in (0, 1)$. Then if the connectivity radius $R$ satisfies [47]*

$$R = \Omega\left(\sqrt{\log^{1+\epsilon}(M)/M}\right), \quad \text{for any } \epsilon > 0, \tag{146}$$

*then with high probability*

$$\xi(\widetilde{\mathcal{L}}) = \mathcal{O}\left(\frac{\log(M)}{M}\right). \tag{147}$$

*Further, from the proof of [56, Lemma 10], for any $\epsilon$ and $c > 0$, if*

$$R = \Omega\left(\sqrt{\log^{1+\epsilon}(M)/(M\pi)}\right) \tag{148}$$

40

*then with probability at least $1 - 2/M^{c-1}$, the following holds*

$$\log^{1+\epsilon} M - \sqrt{2}c \log M \le d_i \le \log^{1+\epsilon} M + \sqrt{2}c \log M, \ \forall \ i. \tag{149}$$

*This means that (124) is satisfied (with $k = 1$) with high probability (also see discussion at the end of [47, Section V]). Therefore using the expression (122) we obtain the following:*

$$\widetilde{C}_2^{\text{geometric}} \times \frac{1}{\sqrt{\xi(\widetilde{\mathcal{L}})}} \le (12500U + 2560) \times \mathcal{O}\left(\frac{\sqrt{M}}{\sqrt{\log(M)}}\right). \tag{150}$$

**Cycle/Path graph.** *Cycle/path graphs satisfy (124) with $k = 2$. We also have $\underline{\lambda}_{\min}(\widetilde{\mathcal{L}}) \ge 1/M^2$ (see the discussion in Sec. 2.3). Therefore using the expression (137) we obtain the following:*

$$\widetilde{C}_2^{\text{cycle}} \times \frac{1}{\sqrt{\xi(\widetilde{\mathcal{L}})}} \le (12500U + 2560) \times M. \tag{151}$$

From the above comparison, it is clear that the rate bounds for xFILTER is about $\mathcal{O}(M)$ times better than the D-GPDA for the path/cycle graph.

We also note that for the xFILTER algorithm, the fact that $L_i = U, \ \forall \ i$ implies that the matrix $\Sigma^2$ given in (125) is a multiple of identity matrix. Therefore by Remark 4.3, we can conclude that in this case xFILTER belongs to both $\mathcal{A}$ and $\mathcal{A}'$.

Now we are ready to present our tightness analysis on D-GPDA and xFILTER.

**Theorem 6.3** *We have the following tightness results.*
**(1)** *Let $D = 1$ and consider the class $(P_U^M, \mathcal{N}_D^M)$. Then D-GPDA is an optimal algorithm, and its convergence rate in (121) is tight (up to a universal constant).*
**(2)** *Let $D = M - 1$ and consider the class $(P_U^M, \mathcal{N}_D^M)$. Then xFILTER is an optimal algorithm, and its convergence rate in (136) is tight (up to a polylog factor).*
**(3)** *More generally, consider the problem class $P_U^M$, and a subclass of $\mathcal{N}_D^M$ satisfying (124). Then the convergence rate in (136) is tight (up to a polylog factor).*

**Proof.** We divide the proof into different cases.
**Case 1).** The network class is a complete graph with $M$ nodes. Using the parameters in (118), $C_2$ is given by (141), and we have that $\Sigma^2 = \frac{80U}{(M-1)M}I_E$. Note that the following holds

$$\|Fx\|^2 = \sum_{(i,j):i \sim j} \|x_i - x_j\|^2.$$

If (114) holds, then Theorem 5.1 and Theorem 6.1 imply

$$T \le 8\left(f(0) - \underline{f} + \frac{2}{MU}\|d_0\|^2\right) \times \frac{400U + 4}{\epsilon}.$$

41

For complete graph it is easy to check that $\xi(\mathcal{G}) \geq 1$. Using the definition in (18), we also have

$$h_T^* = \min_{r \in [T]} \left\| \frac{1}{M} \sum_{i=1}^M \nabla f_i(x_i^r) \right\|^2 + \frac{U}{M^2} \|Ax^r\|^2$$

$$\leq \min_{r \in [T]} \left\| \frac{1}{M} \sum_{i=1}^M \nabla f_i(x_i^r) \right\|^2 + \frac{1}{80} \|\Sigma Ax^r\|^2 \leq e(T) \leq \epsilon.$$

By comparing the lower bound derived in Lemma 3.2, we conclude that the above rate bound is tight (up to some universal constants).

**Case 2).** The network class is a path graph with $M = D + 1$. From Section 2.3 we have

$$\xi(\mathcal{G}) \geq \frac{1}{M^2}. \tag{152}$$

Further we note that condition (124) satisfies with $k = 2$. We have

$$\widetilde{\mathcal{L}} = P^{-1/2} F^T F P^{-1/2} = \mathcal{L}. \tag{153}$$

Therefore we conclude that

$$\xi(\widetilde{\mathcal{L}}) \geq \xi(\mathcal{G}) \geq \frac{1}{M^2}. \tag{154}$$

Applying the above estimate to (125), we can choose

$$\Sigma^2 = \frac{4608U}{4(M-2)\underline{\lambda}_{\min}(\widetilde{\mathcal{L}})} I_E, \quad \Upsilon^2 = \frac{96U}{4(M-2)} P. \tag{155}$$

Using these choices, again we will have

$$\xi(R) \overset{(130)}{\geq} \frac{\xi(\widetilde{\mathcal{L}})}{50} \overset{(154)}{\geq} \frac{1}{50M^2}. \tag{156}$$

Using these constants, and note $D \leq M$, we have

$$h_{T_r}^* = \min_{r \in [T_r]} \left\| \frac{1}{M} \sum_{i=1}^M \nabla f_i(x_i^r) \right\|^2 + \frac{U}{M\underline{\lambda}_{\min}(P^{1/2}\mathcal{L}P^{1/2})} \sum_{(i,j):i \sim j} \|x_i - x_j\|^2$$

$$\leq \min_{r \in [T_r]} \left\| \frac{1}{M} \sum_{i=1}^M \nabla f_i(x_i^r) \right\|^2 + \frac{U}{\underline{\lambda}_{\min}(\mathcal{L})M} \|Fx^r\|^2$$

$$\overset{(155)}{\leq} \min_{r \in [T_r]} \left\| \frac{1}{M} \sum_{i=1}^M \nabla f_i(x_i^r) \right\|^2 + \frac{1}{2304} \|\Sigma Fx^r\|^2 \leq e(T_r),$$

where in the first inequality we have used $P \succeq I_M$. Similarly as in the previous case, suppose $e(T_r) \leq \epsilon$,

then according to Theorem 6.2 we have

$$\epsilon \leq \left( f(0) - \inf_x f(x) + \|d_0\|^2 \frac{5}{MU} I_M \right) \times \frac{128(96U+19)}{T_r}.$$

Recall that for xFILTER, $T_r$ represents the number of times the dual update (80) is performed. Between two dual updates $Q$ primal iterations are performed, where the precise number is given in (102). According to (156) we have

$$\sqrt{1/\xi(R)} \leq 13M. \tag{157}$$

Overall, the total number of iterations required is given by

$$T \leq \frac{1}{\epsilon} \left( f(x^0) - \underline{f} + \frac{5}{MU} \|d_0\|^2 \right) \times 128(96U+19)$$
$$\times \frac{1}{4} \ln \left( \frac{50^2 M^{10} \times (16 + 128M \max\{50 \times 192U, 1\})}{\min\{1, 96^2 \times 4U^2/M^2\}} \right) \times 13M. \tag{158}$$

This implies that the lower bound obtained in Theorem 3.1 is tight up to some universal constant and a ploylog factor in $M$, and the bound-achieving algorithm in class $\mathcal{A}$ is the xFILTER.

**Case 3).** The proof follows similar steps are in the previous case. When $L_i = L_j$, $\forall\, i \neq j$, and when (124) is satisfied, it is easy to verify that the following holds

$$h_{T_r}^* \leq e(T_r), \text{ and } \widetilde{\mathcal{L}} = \mathcal{L}. \tag{159}$$

To bound the total number of iteration required to achieve $h_{T_r}^* \leq \epsilon$, note that when (124) is satisfied, we can apply the bound (136) in Theorem 6.2 and obtain

$$T \leq \frac{1}{\epsilon} \left( f(x^0) - \underline{f} + \frac{5}{MU} \|d_0\|^2 \right) \times 128 \, (96kU+19)$$
$$\times \frac{1}{4} \ln \left( \frac{50^2 M^4 \times (16 + 128M \max\{50 \times 96kU, 1\})}{\xi^3(\mathcal{G}) \times \min\{1, 96^2 k^2 U^2/M^2\}} \right) \sqrt{50/\xi(\mathcal{G})}. \tag{160}$$

Comparing with the lower bound obtained in Theorem 3.1, it is clear that apart from the multiplicative $\ln(\cdot)$ term, the remaining bound is in the same order as the lower bound given in (58). **Q.E.D.**

**Remark 6.4 (Optimal Number of Gradient Evaluations)** *It is important to note that the "outer" iteration of the xFILTER required to achieve $\epsilon$-local solution scales with $\mathcal{O}(U/\epsilon)$, which is independent of the network size. Because local gradient evaluation is only performed in the outer iterations, the above fact suggests that the total number of gradient evaluation required is also in this order, which is optimal because it is the same as what is needed for the centralized gradient descent.*

**Remark 6.5 (Performance Gap Between D-GPDA and xFILTER)** *If we apply D-GPDA to the path or cycle graph, then according to Remark 6.2, the corresponding $C_2$, as well as the final upper bound, will be in $\mathcal{O}(M^2U)$, which is $\mathcal{O}(M)$ worse than the lower bound. Intuitively, this phenomenon happens because of the following: in order to decompose the entire problem into the individual nodes, the x-update* (72a) *has to create a proximal term that matches the quadratic penalty $\|\Sigma Ax\|^2$. But such an additional*

43

*term forces the variables to stay close to their previous iteration. In contrast, xFILTER circumvents the above difficulty by leaving the quadratic penalty intact, but instead using a few fast and decomposable iterations to approximately solve the resulting problem.*

**Remark 6.6 (An Alternative Bound)** *For problems and graphs in $(\mathcal{P}_U^M, \mathcal{N}_D^M)$ without additional conditions, it can be verified that the second choice of the parameters (131) gives the following convergence rates [cf. (138)]*

$$T = \widetilde{\mathcal{O}}\left(\left(f(0) - \inf_x f(x) + \|d_0\|^2 \frac{5}{MU} I_M\right) \times \frac{U}{\epsilon} \times \frac{1}{\sqrt{\xi(P^{1/2}\mathcal{L}P^{1/2})}}\right), \tag{161}$$

*where the notation $\widetilde{\mathcal{O}}$ denotes $\mathcal{O}$ with a multiplicative ploylog factor. The above rate is proportional to the square root of the eigengap for the matrix $P^{1/2}\mathcal{L}P^{1/2}$, which is the unnormalized Laplacian matrix for graph $\mathcal{G}$.*

**Remark 6.7 (Non-uniform Lipschitz Constants)** *We comment that for the general case $L_i \neq L_j$, $\forall\, i, j$, we can use similar steps to verify that the bound (138) derived in Theorem 6.2 is optimal, in the sense that they achieve the lower bound (65) predicted in Corollary 3.2.*

# 7 Numerical Results

This section presents numerical examples to show the effectiveness of the proposed algorithms. Two kinds of problems are considered, distributed binary classification and distributed neural networks training. We use the former one to demonstrate the behavior and scalability of our algorithm and use the latter one to show the practical performance.

## 7.1 Simulation Setup

In our simulations, all algorithms are implemented in MATLAB R2017a for binary classification problem and implemented in Python 3.6 for training neural networks, running on a computer node with two 12-core Intel Haswell processors and 128 GB of memory (unless otherwise specified). Both synthetic and real data are used for performance comparison. For synthetic data, the feature vector is randomly generated with standard normal distribution with zero mean and unit variance. The label vector is randomly generated with uniformly distributed pseudorandom integers taking the values $\{-1, 1\}$. For real data, we use the breast cancer dataset [1] for binary classification and MNIST[2] for training neural network. The breast cancer dataset contains a total of 569 samples each with 30 real positive features. The MNIST dataset contains a total of 60,000 handwritten digits, each with a $28 \times 28$ gray scale image and a label from ten categories.

---

[1]https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)
[2]http://yann.lecun.com/exdb/mnist/

**Figure 6:** $M = 5, B = 200, K = 10$



**Figure 7:** $M = 10, B = 200, K = 10$



**Figure 8:** $M = 20, B = 200, K = 10$



**Figure 9:** $M = 20, B = 50, K = 10$



**Figure 10:** $M = 20, B = 100, K = 10$



**Figure 11:** $M = 20, B = 400, K = 10$

## 7.2 Distributed Binary Classification

We consider a non-convex distributed binary classification problem [57]. The global consensus problem (2) can be expressed as follows:

$$\min_{x \in \mathbb{R}^{SM}} \; f(x) := \frac{1}{M} \sum_{i=1}^{M} f_i(x_i), \quad \text{s.t. } x_i = x_j, \forall \, (i,j) \in \mathcal{E}.$$

And each component function $f_i$ is expressed by

$$f_i(x_i) = \frac{1}{B} \sum_{j=1}^{B} \log \left( 1 + \exp(-y_{ij} x_i^T v_{ij}) \right) + \sum_{s=1}^{S} \frac{\lambda \alpha x_{i,s}^2}{1 + \alpha x_{i,s}^2}.$$

Here $v_{ij} \in \mathbb{R}^S$ denotes the feature vector with dimension $S$, $y_{ij} \in \{1, -1\}$ denotes the label for the $j$th date point in $i$th agent, and there are total $B$ data points for each agent. Unless otherwise noted, the graph $\mathcal{E}$ used in our simulation is generated using the random geometric graph and the graph parameter $Ra$ is set to 0.5. The regularization parameter is set to $\lambda = 0.001, \alpha = 1$.

To compare the convergence performance of the proposed algorithms, we randomly generated $MB$ data points with dimension $K$ and distribute them into $M$ nodes, i.e. each node contains $B$ data points with $K$ features. Then we compare the proposed xFILTER and D-GPDA with the distributed subgradient (DSG) method [4], the Push-sum algorithm [58], and the NEXT algorithm [13]. The parameters for NEXT are

**Figure 12:** $M = 10, B = 20, K = 5$



**Figure 13:** $M = 10, B = 20, K = 10$



**Figure 14:** $M = 10, B = 20, K = 20$



**Figure 15:** $M = 50, B = 2000, K = 10$



**Figure 16:** $M = 10, B = 56, K = 30$

chosen as $\tau = 1, \alpha[0] = 0.1$ and $\mu = 0.01$ as suggested by [13], while the parameters for xFILTER are chosen based on (125).

Simulation results on synthetic data for different $M, B, K$ averaged over 30 realizations are investigated and shown in Fig. 6 to Fig. 15, where the x-axis denotes the total rounds of communications required, and the y-axis denotes the quality measure (18) proposed in Section 2. Note that the curves xFILTER (outer) included in these figures show the number of communication rounds required for xFILTER to perform the "outer" iterations (which is equivalent to $r$ in Algorithm 2, since in each outer iteration only one round of communication is required in Step **S3**). The performance evaluated on real data is also characterized in Fig. 16, in which we choose $M = 10$, $B = 56$, and $K = 30$. These results show that the proposed algorithms perform well in all parameter settings compared with existing methods.

We further note that these figures also show (rough) comparison about computation efficiency of different algorithms. Specifically, for D-GPDA, DSG and Push Sum (resp. NEXT), the total rounds of communication is the same as (resp. twice as) the total number of gradient evaluations per node. In contrast, the total rounds of communication in the *outer loop* of xFILTER is the same as the local gradient evaluations. Therefore, the comparison between xFILTER (outer) and other algorithms in Fig. 8 to Fig. 15 shows the relative computational efficiency of these algorithms. Clearly, xFILTER has a significant

(a) $B = 10, K = 10, \epsilon = 10^{-10}$     (b) $B = 200, K = 10, \epsilon = 10^{-15}$

**Figure 17:** Comparison of NEXT and xFILTER over path graphs with increasing number of nodes ($M \in [10, 150]$ in (a) and $M \in [5, 50]$ in (b)). Each point in the figure represents the total number of communication needed to reach $h_T^* \leq \epsilon$.

**Table 2:** Optimality gap after 200 rounds of communications ($B = 200, K = 10$)

| number of nodes $M$ | D-GPDA | xFILTER |
|:---:|:---:|:---:|
| 10 | $3.96 \times 10^{-4}$ | $2.50 \times 10^{-11}$ |
| 20 | $5.45 \times 10^{-4}$ | $1.92 \times 10^{-9}$ |
| 30 | $1.20 \times 10^{-4}$ | $4.71 \times 10^{-11}$ |
| 40 | $2.95 \times 10^{-4}$ | $4.07 \times 10^{-10}$ |
| 50 | $3.88 \times 10^{-4}$ | $8.47 \times 10^{-11}$ |

advantage over the rest of the algorithms.

Further, we compare the scalability performance of the proposed algorithms with increased network dimension $M$, and the results are shown in Fig. 17, Table 2 and Table 3. In particular, in Fig. 17 we compare the total communication rounds required for NEXT and the xFILTER for reaching $h_T^* \leq 10^{-10}$ and $h_T^* \leq 10^{-15}$, over path graphs with increasing number of nodes. Overall, we see that the xFILTER performs reasonably fast.

We do want to point out that although for the unconstrained problems that we have tested, our proposed algorithms compare relatively favorably with NEXT, NEXT can in fact handle a larger class of problems because it is designed for nonsmooth and constrained nonconvex problems. Further, for all the algorithms we have used, we did not tune the parameters: For xFILTER and D-GPDA, we use the theoretical upper bound suggested in Theorem 5.1, and for NEXT we use the parameters suggested in the paper [13]. For all our tested problems and algorithms, it is possible to fine-tune the stepsizes to make them faster, but since this paper is mostly on the theoretical properties of rate optimal algorithms, we choose not to go down that path.

## 7.3   Distributed Neural Network Training

In our second experiment, we present some numerical results under a more realistic setting. We consider training a neural network model for fitting the MNIST data set. The dataset is first randomly partitioned

47

**Table 3:** Optimality gap after 1000 rounds of communications ($B = 200, K = 10$)

| number of nodes $M$ | D-GPDA | xFILTER |
|:---:|:---:|:---:|
| 10 | $8.24 \times 10^{-13}$ | $1.93 \times 10^{-33}$ |
| 20 | $9.41 \times 10^{-12}$ | $1.43 \times 10^{-32}$ |
| 30 | $2.09 \times 10^{-13}$ | $2.26 \times 10^{-32}$ |
| 40 | $1.52 \times 10^{-11}$ | $4.19 \times 10^{-33}$ |
| 50 | $2.30 \times 10^{-10}$ | $6.48 \times 10^{-33}$ |



(a) Training Loss

(b) Training Accuracy

**Figure 18:** Comparison of DSG and xFILTER over path graphs on distributed training neural networks; Plot (a) shows the dynamic of the categorical cross-entropy loss, and plot (b) shows the training classification accuracy. The parameters are chosen based on their best practical performance through grid search. The curves xFILTER (outer) and xFILTER (total) again represent the number of outer iteration, and the total number of iterations required for xFILTER.

into 10 subsets, and then gets distributed over 10 machines. A fully connected neural network with one hidden layer is used in the experiment. The number of neurons for the hidden layer and the output layer are set as 128 and 10, respectively. The initial weights for the neural network are drawn from a truncated normal distribution centered at zero with variance scaled with the number of input units. The algorithms are written in Python, and the communication protocol is implemented using the Message Passing Interface (MPI). The empirical performance of the xFILTER is evaluated and compared with the DSG algorithm [59]. Fig. 18 shows that, compared with DSG, the proposed algorithm achieves better communication and computation efficiency, and has improved classification accuracy.

Note that despite the fact that some global parameters (such as the Lipschitz constants) are unknown, the rules provided in (125) or (131) still can help us roughly estimate a set of good parameters. For example, we choose the following parameters

$$\Sigma^2 = \frac{\sigma}{\sum_i d_i \lambda_{\min}(\widetilde{\mathcal{L}})}, \quad \Upsilon^2 = \frac{\beta P}{\sum_i d_i}, \tag{162}$$

and tune the parameter $\beta$ and $\sigma$ by searching from the set $\{0.1, 0.2, 0.5, 1, 2, 5, \cdots, 100, 200, 500\}$. Based on the best practical performance over 10 runs, we choose $\beta = 100$ and $\sigma = 20$ for xFILTER and $\alpha = 0.1$ for DSG.

**Figure 19:** Graphical comparison of various bounds analyzed in this work, illustrated over a path graph with $M$ nodes.

# 8   Conclusion and Future Works

This paper represents the first work that investigates the performance of optimal first-order non-convex algorithms for distributed information processing and optimization problems. We first set our scope by defining the problem, network, and algorithm classes $(\mathcal{P}, \mathcal{N}, \mathcal{A})$ that are under consideration. We then provide a lower complexity bound that characterizes the worst case performance for any first-order distributed algorithm in class $\mathcal{A}$, and finally propose and analyze two algorithms that are capable of (nearly) achieving the lower bound in various settings. The various bounds discussed in the work is illustrated in Fig. 19 through a $M$-node path graph as an example.

To the best of our knowledge, the proposed algorithms are the first and the only available distributed non-convex algorithms in class $\mathcal{A}$ that can optimally reduce *both* the size of the gradient and the consensus error for $(\mathcal{P}, \mathcal{N})$, and achieving the (near) optimal rate performance for problem/network classes $(\mathcal{P}, \mathcal{N})$. However, they still require some global information to initialize the parameters, so it will be of interest to design global information free algorithms that only require local structures to set parameters (just like in the convex case, see discussions in [60]). It will also be desirable to merge the inner Chebyshev iteration with the outer dual update to design a *single-loop* algorithm, and to extend the proposed algorithms to problems with nonsmooth regularizers and constraints.

# 9   Appendix

## 9.1   Proof of Lemma 5.1

**Proof.** First we show that for all $r \geq -1$ the following holds for D-GPDA

$$\nabla f(x^r) + F^T \lambda^r + F^T \Sigma^2 F x^{r+1} + H(x^{r+1} - x^r) = 0. \tag{163}$$

Note that for the initialization (70) we have

$$\nabla f(x^{-1}) + (2\Delta + \Upsilon^2)x^0 = \nabla f(x^{-1}) + (F^T \Sigma^2 F + H)x^0 = 0.$$

49

Setting $x^{-1} = 0, \lambda^{-1} = 0$ and using (70), we obtain

$$\nabla f(0) + F^T \lambda^{-1} + F^T \Sigma^2 F x^0 + H(x^0 - x^{-1}) = 0. \tag{164}$$

Further, the optimality condition of the $x$ update (72a) suggests that (163) holds for all $r \geq 0$, therefore (163) is proved.

Second, by using (163) and the $y$ update (72b), we obtain

$$F^T \lambda^{r+1} = -\nabla f(x^r) - H(x^{r+1} - x^r), \ \forall \ r \geq -1. \tag{165}$$

Then subtracting the previous iteration leads to

$$F^T(\lambda^{r+1} - \lambda^r) = -(\nabla f(x^r) - \nabla f(x^{r-1})) - Hw^{r+1}, \ \forall \ r \geq 0.$$

Note that the matrix $H \succ 0, \Sigma^2 \succ 0$, then we have

$$H^{-1/2}(\Sigma F)^T \Sigma^{-1}(\lambda^{r+1} - \lambda^r) = -H^{-1/2}(\nabla f(x^r) - \nabla f(x^{r-1})) - H^{1/2} w^{r+1}. \tag{166}$$

Then using the fact that

$$\Sigma^{-1}(\lambda^{r+1} - \lambda^r) = \Sigma F x^{r+1} \in \text{col}(\Sigma F),$$

we can square both sides and obtain the following

$$\begin{aligned}
&\underline{\lambda}_{\min}(\Sigma F H^{-1} F^T \Sigma) \|\Sigma^{-1}(\lambda^{r+1} - \lambda^r)\|^2 \\
&\leq 2\|H^{-1/2}(\nabla f(x^r) - \nabla f(x^{r-1}))\|^2 + 2(w^{r+1})^T H w^{r+1} \\
&\leq 2\|\Upsilon^{-1}(\nabla f(x^r) - \nabla f(x^{r-1}))\|^2 + 2(w^{r+1})^T H w^{r+1} \\
&\stackrel{(20)}{\leq} \frac{2}{M^2}\|\Upsilon^{-1} L(x^r - x^{r-1})\|^2 + 2\|w^{r+1}\|_H^2, \ \forall \ r \geq 0.
\end{aligned} \tag{167}$$

This concludes the proof of the first part.

To show the second part, note that according to (82b), $x^{r+1}$ generated by xFILTER is given by (for all $r \geq -1$)

$$\nabla f(x^r) + F^T(\lambda^r + \Sigma^2 F x^{r+1}) + \Upsilon^2(x^{r+1} - x^r) = \Upsilon^2 R \epsilon^{r+1}. \tag{168}$$

Then use the same analysis steps as in the first part, we arrive at the desired result.     **Q.E.D.**

## 9.2 Proof of Lemma 5.2

**Proof.** Using the Lipschitz gradient assumption (20), we have

$$
\begin{aligned}
\mathsf{AL}(x^{r+1}, \lambda^r) - \mathsf{AL}(x^r, \lambda^r) &\leq \langle \nabla f(x^r) + f^T \lambda^r + F^T \Sigma^2 F x^r, x^{r+1} - x^r \rangle \\
&\quad + \frac{1}{2M} \|x^{r+1} - x^r\|_L^2 + \frac{1}{2} \|\Sigma F(x^{r+1} - x^r)\|^2 \\
&= \langle \nabla f(x^r) + F^T \lambda^r + A^T \Sigma^2 F x^{r+1}, x^{r+1} - x^r \rangle \\
&\quad + \langle H(x^{r+1} - x^r), x^{r+1} - x^r \rangle + \frac{1}{2M} \|x^{r+1} - x^r\|_L^2 \\
&\quad + \frac{1}{2} \|\Sigma F(x^{r+1} - x^r)\|^2 - \|x^{r+1} - x^r\|_{H + F^T \Sigma^2 F} \\
&\overset{(163),(22)}{\leq} -(x^{r+1} - x^r)^T \left( \frac{\Delta}{2} - \frac{L}{2M} + \Upsilon^2 \right) (x^{r+1} - x^r). \quad\quad (169)
\end{aligned}
$$

Using the update rule of the dual variable, and combine the above inequality, we obtain

$$
\begin{aligned}
\mathsf{AL}^{r+1} - \mathsf{AL}^r &\leq -\frac{1}{2} \|x^{r+1} - x^r\|_{\Delta + 2\Upsilon^2 - L/M}^2 + \langle \lambda^{r+1} - \lambda^r, A x^{r+1} \rangle \\
&= -\frac{1}{2} \|x^{r+1} - x^r\|_{\Delta + 2\Upsilon^2 - L/M}^2 + \|\Sigma^{-1}(\lambda^{r+1} - \lambda^r)\|^2
\end{aligned}
$$

Combined with Lemma 5.1 we complete the first part.

The second part follows similar steps. The modifications are that $H$ is replaced by $\Upsilon^2$, and that there is an additional error term in the optimality condition; cf. (82b).       **Q.E.D.**

## 9.3 Proof of Lemma 5.5 and Lemma 5.6

**Proof.** Using the optimality condition (163), we have

$$
\langle F^T \lambda^{r+1} + \nabla f(x^r) + H(x^{r+1} - x^r), x^{r+1} - x \rangle = 0, \ \forall \ r \geq -1
$$

This implies that for all $r \geq 0$

$$
\langle F^T(\lambda^{r+1} - \lambda^r) + \nabla f(x^r) - \nabla f(x^{r-1}) + H w^{r+1}, x^{r+1} - x^r \rangle = 0.
$$

It follows that

$$
\begin{aligned}
\frac{1}{2} \|\Sigma F x^{r+1}\|^2 + \frac{1}{2} \|x^{r+1} - x^r\|_H^2 &\leq \frac{1}{2} \|\Sigma F x^r\|^2 + \frac{1}{2} \|x^r - x^{r-1}\|_H - \frac{1}{2} \|w^{r+1}\|_H^2 \quad\quad (170) \\
&\quad + \frac{1}{2M} \|x^{r+1} - x^r\|_L^2 + \frac{1}{2M} \|x^r - x^{r-1}\|_L^2, \quad \forall \ r \geq 0.
\end{aligned}
$$

Then combining Lemma 5.2 and (170), for all $r \geq 0$ we have

$$
\begin{aligned}
P^{r+1} - P^r &\leq -\left( \frac{c}{2} - 2\kappa \right) \|w^{r+1}\|_H^2 \\
&\quad - \frac{1}{2}(x^{r+1} - x^r)^T \left( \Delta + 2\Upsilon^2 - \frac{L}{M} - \frac{4\kappa}{M^2} L \Upsilon^{-2} L - \frac{2cL}{M} \right) (x^{r+1} - x^r).
\end{aligned}
$$

Therefore, in order to make the potential function decrease, we need to follow (108).

To show a similar result for the xFILTER, consider the following optimality condition derived from (168)

$$\langle F^T \lambda^{r+1} + \nabla f(x^r) + \Upsilon^2(x^{r+1} - x^r) - \Upsilon^2 R \epsilon^{r+1}, x^{r+1} - x \rangle = 0, \ \forall \ x.$$

Following similar steps as in (170), and use (103), we have

$$\frac{1}{2}\|\Sigma F x^{r+1}\|^2 + \frac{1}{2}\|x^{r+1} - x^r\|_{\Upsilon^2}^2 \leq \frac{1}{2}\|\Sigma F x^r\|^2 + \frac{1}{2}\|x^r - x^{r-1}\|_{\Upsilon^2} - \frac{1}{2}\|w^{r+1}\|_{\Upsilon^2}^2 \tag{171}$$
$$+ 1/(2M)\|x^{r+1} - x^r\|_L^2 + 1/(2M)\|x^r - x^{r-1}\|_L^2$$
$$+ 1/4\|x^{r+1} - x^r\|_{\Upsilon^2 R}^2 + 1/4\|x^r - x^{r-1}\|_{\Upsilon^2 R}^2, \quad \forall \ r \geq -1.$$

Then combining Lemma 5.2, (170), and the estimate of the size of $\epsilon$ in (103), we have

$$\widetilde{P}^{r+1} - \widetilde{P}^r \leq -\frac{1}{2}(x^{r+1} - x^r)^T V(x^{r+1} - x^r) - \left(\frac{\widetilde{c}}{2} - 3\widetilde{\kappa}\right)\|w^{r+1}\|_{\Upsilon^2}^2.$$

with

$$V := \left(\Upsilon^2 R - (1 + 2\widetilde{c})\frac{L}{M} - \frac{6\widetilde{\kappa}}{M^2}L\Upsilon^{-2}L - \frac{\Upsilon^2 R(24\widetilde{\kappa} + 6 + 16\widetilde{c})}{16}\right).$$

Therefore in order to make the potential function decrease, we need to follow (110). **Q.E.D.**

## 9.4 Proof of Lemma 5.7

**Proof.** For D-GPDA, we can express the AL as (for all $r \geq 0$)

$$\mathsf{AL}^{r+1} - f(x^{r+1}) = \langle \lambda^{r+1}, \Sigma^{-2}(\lambda^{r+1} - \lambda^r)\rangle + \frac{1}{2}\|\Sigma F x^{r+1}\|^2$$
$$= \frac{1}{2}\left(\|\Sigma^{-1}\lambda^{r+1}\|^2 - \|\Sigma^{-1}\lambda^r\|^2 + \|\Sigma^{-1}(\lambda^{r+1} - \lambda^r)\|^2 + \|\Sigma F x^{r+1}\|^2\right).$$

Since $\inf_x f(x) = \underline{f}$ is lower bounded, let us define

$$\widehat{\mathsf{AL}}^{r+1} := \mathsf{AL}^{r+1} - \underline{f}, \ \widehat{f}(x) := f(x) - \underline{f} \geq 0, \ \widehat{P}^{r+1} := P^{r+1} - \underline{f}.$$

Therefore, summing over $r = -1 \cdots, T$, we obtain

$$\sum_{r=-1}^{T} \widehat{\mathsf{AL}}^{r+1} = \frac{1}{2}\left(\|\Sigma^{-1}\lambda^{T+1}\|^2 - \|\Sigma^{-1}\lambda^{-1}\|^2\right)$$
$$+ \sum_{r=-1}^{T} \left(\widehat{f}(x^{r+1}) + \frac{1}{2}\|\Sigma F x^{r+1}\|^2 + \frac{1}{2}\|\Sigma^{-1}(\lambda^{r+1} - \lambda^r)\|^2\right).$$

Using the initialization $\lambda^{-1} = 0$, then the above sum is lower bounded by zero. This fact implies that the sum of $\widehat{P}^{r+1}$ is also lower bounded by zero (note, the remaining terms in the potential function are all

nonnegative)

$$\sum_{r=0}^{T} \widehat{P}^{r+1} \geq 0, \quad \forall\, T > 0,$$

Note that if the parameters of the system are chosen according to (108), then $P^{r+1}$ is nonincreasing, which implies that its shifted version $\widehat{P}^{r+1}$ is also nonincreasing. Combined with the nonnegativity of the sum of the shifted potential function, we can conclude that

$$\widehat{P}^{r+1} \geq 0, \quad \text{and} \quad P^{r+1} \geq \inf f(x), \quad \forall\, r \geq 0. \tag{172}$$

Next we compute $P^0$. By using (22), we have

$$P^0 = \mathsf{AL}^0 + \frac{2\kappa}{M^2}\|\Upsilon^{-1}Lx^0\|^2 + \frac{c}{2}\left(\|x^0\|^2_{2\Delta+\Upsilon^2+L/M}\right) \tag{173}$$

$$\mathsf{AL}^0 \leq f(x^0) + 2\|\Sigma F x^0\|^2$$

$$x^0 \overset{(70)}{=} (2\Delta + \Upsilon^2)^{-1}\frac{1}{M}[\nabla f_1(0); \cdots ; \nabla f_M(0)]$$

$$= (2\Delta + \Upsilon^2)^{-1}\frac{1}{M}d_0 \tag{174}$$

where in the last equality we have used the definition of $d_0$ in (48). Use the above relations, we have

$$P^0 \leq f(x^0) + (x^0)^T Z x^0 \tag{175}$$

with the matrix $Z$ defined as

$$Z = \frac{2\kappa}{M^2}L\Upsilon^{-2}L + \frac{c(2\Delta + \Upsilon^2 + L/M)}{2} + 2F\Sigma^2 F \preceq 4c(2\Delta + \Upsilon^2)$$

where the last inequality follows from our choice of parameters in (108b), and the fact $c \geq 1$. Note that

$$4c\|x^0\|^2_{2\Delta+\Upsilon^2} \leq \frac{4c}{M^2}d_0^T(2\Delta + \Upsilon^2)^{-1}(2\Delta + \Upsilon^2)(2\Delta + \Upsilon^2)^{-1}d_0$$

$$= \frac{4c}{M^2}d_0^T(2\Delta + \Upsilon^2)^{-1}d_0 \leq \frac{2}{M}d_0^T L^{-1}d_0 \tag{176}$$

where the last inequality comes from the choice of the parameters (108b), which implies that $2\Delta+\Upsilon^2 \succeq 2\frac{Lc}{M}$. These constants combined with (173) shows the desired result.

For xFILTER, the proof for the lower boundedness is the same. To bound the size of $\widetilde{P}^0$, first note that we again have

$$\mathsf{AL}^{r+1} - f(x^{r+1}) = \frac{1}{2}\big(\|\Sigma^{-1}\lambda^{r+1}\|^2 - \|\Sigma^{-1}\lambda^r\|^2 + \|\Sigma^{-1}(\lambda^{r+1} - \lambda^r)\|^2 + \|\Sigma F x^{r+1}\|^2\big).$$

By letting $r = -1$, and use the fact that $x^{-1} = 0$ and $\lambda^{-1} = 0$, we obtain

$$\mathsf{AL}^0 - f(x^0) = \frac{1}{2}\big(2\|\Sigma^{-1}\lambda^0\|^2 + \|\Sigma F x^0\|^2\big) = \frac{3}{2}\|\Sigma^{-1}\lambda^0\|^2. \tag{177}$$

Then we have

$$\widetilde{P}^0 = \mathsf{AL}^0 + \frac{3\widetilde{\kappa}}{M^2}\|\Upsilon^{-1}Lx^0\|^2 + \frac{3}{8}\widetilde{\kappa}\|x^0\|^2_{\Upsilon^2 R}$$
$$+ \frac{\widetilde{c}}{2}\left(\|\Sigma Fx^0\|^2 + \|x^0\|^2_{\Upsilon^2 + \Upsilon^2 R/4 + L/M}\right), \tag{178}$$

$$\mathsf{AL}^0 \le f(x^0) + 2\|\Sigma Fx^0\|^2, \ x^{-1} = 0, \ \lambda^{-1} = 0, \tag{179}$$

$$x^0 \overset{(82b)}{=} R^{-1}\Upsilon^{-2}\nabla f(0) - \epsilon^0, \quad \widetilde{\epsilon}^{-1} \overset{(99)}{=} R^{-1}\Upsilon^{-2}\nabla f(0). \tag{180}$$

Use the above relation, we have

$$\widetilde{P}^0 \le f(x^0) + (x^0)^T \widetilde{Z}x^0 \tag{181}$$

with the matrix $Z$ defined as

$$\widetilde{Z} = \frac{3\widetilde{\kappa}}{M^2}L\Upsilon^{-2}L + \left(\frac{3}{8}\widetilde{\kappa} + \widetilde{c}\right)\Upsilon^2 R + \frac{\widetilde{c}L}{2M} + 2F\Sigma^2 F \preceq 3\Upsilon^2 R$$

where the last inequality follows from our choice of parameters in (110b). Therefore we have

$$(x^0)^T \widetilde{Z}x^0 \le 3(x^0)^T \Upsilon^2 Rx^0$$
$$\le 3(\nabla f(0) - \Upsilon^2 R\epsilon^0)^T R^{-1}\Upsilon^{-2}(\nabla f(0) - \Upsilon^2 R\epsilon^0)$$
$$\overset{(i)}{\le} 6(\nabla f(0))^T R^{-1}\Upsilon^{-2}\nabla f(0) + 6(\epsilon^0)^T \Upsilon^2 R\epsilon^0$$
$$\le 3M(\nabla f(0))^T L^{-1}\nabla f(0) + \frac{3}{8M}\|x^0\|^2_{\Upsilon^2 R} \tag{182}$$

where in (i) we have used the Cauchy-Swartz inequality; the last inequality uses (103), the choice of the parameters (110b) (which implies $\Upsilon^2 R \ge 4L/M$). The above series of inequalities imply that

$$2\|x^0\|^2_{\Upsilon^2 R} \le \left(3 - \frac{3}{8M}\right)\|x^0\|^2_{\Upsilon^2 R} \le 3M(\nabla f(0))^T L^{-1}\nabla f(0).$$

Therefore overall we have

$$(x^0)^T \widetilde{Z}x^0 \le 3(x^0)^T \Upsilon^2 Rx^0 \le 5M(\nabla f(0))^T L^{-1}\nabla f(0). \tag{183}$$

Finally, by observing $\frac{1}{M^2}d_0^T d_0 = \|\nabla f(0)\|^2$, the desired result is obtained. **Q.E.D.**

## 9.5   Proof of Theorem 5.1

**Proof.** To show the first part, we consider the optimality condition (165), and multiply both sides of it by the all one vector, and use the fact that $1^T A^T = 0$ to obtain

$$1^T \nabla f(x^r) + 1^T H(x^{r+1} - x^r) = 0, \ \forall \ r \ge -1.$$

Squaring both sides and rearranging terms, we have

$$\left\|\frac{1}{M}\sum_{i=1}^{M}\nabla f_i(x_i^r)\right\|^2 \le (x^{r+1}-x^r)^T H11^T H(x^{r+1}-x^r)$$

$$\le (x^{r+1}-x^r)^T H(x^{r+1}-x^r) \times 1^T H1$$

$$\le \|x^{r+1}-x^r\|_H^2 \times \left(4\sum_{(i,j)i\sim j}\sigma_{ij}^2 + \sum_{i=1}^{M}\beta_i^2\right), \forall\, r \ge -1.$$

Combining with (109), we obtain, for all $r \ge 0$

$$\left\|\frac{1}{M}\sum_{i=1}^{M}\nabla f_i(x_i^r)\right\|^2 \le \|x^{r+1}-x^r\|_H^2\left(4\sum_{e\in\mathcal{E}}\sigma_e^2 + \sum_{i=1}^{M}\beta_i^2\right)$$

$$\le 8\left(P^r - P^{r+1}\right)\left(4\sum_{e\in\mathcal{E}}\sigma_e^2 + \sum_{i=1}^{M}\beta_i^2\right). \tag{184}$$

where in the last inequality we used $2(\Delta + \Upsilon^2) \succeq H$. We then bound the consensus error. Lemma 5.1 implies

$$\|\Sigma F x^{r+1}\|^2 \le \kappa\left(\frac{2}{M^2}\|\Upsilon^{-1}L(x^r - x^{r-1})\|^2 + 2\|w^{r+1}\|_H^2\right)$$

$$\overset{(108a)}{\le} 2\kappa\left(4\|w^{r+1}\|_H^2 + \frac{2}{M^2}\|\Upsilon^{-1}L(x^{r+1}-x^r)\|^2\right). \tag{185}$$

Therefore

$$\|\Sigma F x^r\|^2 \le 4\kappa\left(4\|w^{r+1}\|_H^2 + \frac{2}{M^2}\|\Upsilon^{-1}L(x^{r+1}-x^r)\|^2\right) + 2\|\Sigma F(x^{r+1}-x^r)\|^2. \tag{186}$$

Combining with (109), and using the fact that [cf. (108b)]

$$\Delta + \Upsilon^2 \succeq \frac{8\kappa L\Upsilon^{-2}L}{M^2}, \quad 2\Delta \succeq F\Sigma^2 F \tag{187}$$

we have

$$\|\Sigma F x^r\|^2 \le 16\kappa\|w^{r+1}\|_H^2 + 5\|x^{r+1}-x^r\|_{\Delta+\Upsilon^2}^2 \overset{(109)}{\le} 20(P^r - P^{r+1}). \tag{188}$$

Also note that by the definition of $e(T)$ we have

$$T \times e(T) \le \sum_{r=1}^{T}\left(\|\Sigma F x^r\|^2 + \left\|\frac{1}{M}\sum_{i=1}^{M}\nabla f_i(x_i^r)\right\|^2\right) \tag{189}$$

Then combining the above with (109) and (188), and the fact that the potential function is lower bounded by $\underline{f}$, we obtain the desired result.

55

To show the result for xFILTER, multiply both sides of the optimality condition (168) by the all one vector, and use the fact that $F1 = 0$ to obtain

$$1^T \nabla f(x^r) + 1^T \Upsilon^2 (x^{r+1} - x^r) = 1^T \Upsilon^2 R \epsilon^{r+1}. \tag{190}$$

Squaring both sides and rearranging terms we have

$$\left\| \frac{1}{M} \sum_{i=1}^{M} \nabla f_i(x_i^r) \right\|^2 \leq 2(x^{r+1} - x^r)^T \Upsilon^2 11^T \Upsilon^2 (x^{r+1} - x^r) + 2(\epsilon^{r+1})^T \Upsilon^2 R 11^T \Upsilon^2 R \epsilon^{r+1}$$

$$\overset{(103)}{\leq} 2(x^{r+1} - x^r)^T \Upsilon^2 (x^{r+1} - x^r) \times 1^T \Upsilon^2 1 + M/(4M) \|x^{r+1} - x^r\|_{\Upsilon^2 R}^2$$

$$\leq \|x^{r+1} - x^r\|_{\Upsilon^2 R}^2 \times 2 \left( 1 + \sum_{i=1}^{M} \beta_i^2 \right), \ \forall \ r \geq -1.$$

where in the last inequality we have used the fact that $\Upsilon^2 R = \Upsilon^2 + F^T \Sigma^2 F \succeq \Upsilon^2$.

To bound the consensus error, we first use (103) and obtain

$$\|\Upsilon^2 R(\epsilon^{r+1} - \epsilon^r)\|^2 \leq \frac{1}{4M} \|x^{r+1} - x^r\|_{\Upsilon^2 R}^2 + \frac{1}{4M} \|x^r - x^{r-1}\|_{\Upsilon^2 R}^2.$$

Similarly as the first part, we use Lemma 5.1 and obtain

$$\|\Sigma F x^{r+1}\|^2$$

$$\leq 3\widetilde{\kappa} \left( \|x^{r+1} - x^r\|_{\frac{\Upsilon^2 R}{4M}}^2 + \|w^{r+1}\|_{\Upsilon^2}^2 + \|x^r - x^{r-1}\|_{\frac{\Upsilon^2 R}{4M} + \frac{L\Upsilon^{-2}L}{M^2}}^2 \right)$$

$$\leq 2\|x^{r+1} - x^r\|_{\Upsilon^2 R}^2 + 3\widetilde{\kappa}\|w^{r+1}\|_{\Upsilon^2}^2 + 2\|x^r - x^{r-1}\|_{\Upsilon^2 R}^2, \quad \forall \ r \geq 0 \tag{191}$$

where the last inequality comes from (110b), that

$$2\Upsilon^2 R \succeq 3\widetilde{\kappa} \left( \frac{L\Upsilon^{-2}L}{M^2} + \Upsilon^2 R \right). \tag{192}$$

By combining (191) and the following inequality

$$\|\Sigma F x^r\|^2 \leq 2\|\Sigma F(x^{r+1} - x^r)\|^2 + 2\|\Sigma F x^{r+1}\|^2,$$

we have

$$\|\Sigma F x^r\|^2 \leq 4\|x^{r+1} - x^r\|_{\Upsilon^2 R + F^T \Sigma^2 F}^2 + 6\widetilde{\kappa}\|w^{r+1}\|_{\Upsilon^2}^2 + 4\|x^r - x^{r-1}\|_{\Upsilon^2 R}^2$$

$$\overset{(111)}{\leq} 64(\widetilde{P}^r - \widetilde{P}^{r+1}) + 64(\widetilde{P}^{r-1} - \widetilde{P}^r), \quad \forall \ r \geq 1$$

$$\|\Sigma F x^0\|^2 \leq 64(\widetilde{P}^0 - \widetilde{P}^1) + 4\|x^0\|_{\Upsilon^2 R}^2.$$

56

So overall we have that

$$\sum_{r=0}^{T_r} \left( \left\| \frac{1}{M} \sum_{i=1}^{M} \nabla f_i(x_i^r) \right\|^2 + \|\Sigma F x^r\|^2 \right)$$

$$\leq 64 \left( 1 + \sum_{i=1}^{M} \beta_i^2 + 1 \right) \sum_{r=1}^{T_r} ((\widetilde{P}^r - \widetilde{P}^{r+1}) + (\widetilde{P}^{r-1} - \widetilde{P}^r))$$

$$+ 64(\widetilde{P}^0 - \widetilde{P}^1) + 4\|x^0\|_{\Upsilon^2 R}^2$$

$$\leq 128 \left( 1 + \sum_{i=1}^{M} \beta_i^2 + 2 \right) (\widetilde{P}^0 - \underline{f}) + 4\|x^0\|_{\Upsilon^2 R}^2. \tag{193}$$

where the last inequality utilizes the descent property of $\widetilde{P}^0$ in Lemma 5.6. Note that from (177), (178) and use $\widetilde{c} = 8\widetilde{\kappa}$ in (110a), we obtain

$$\widetilde{P}^0 \geq f(x^0) + \widetilde{\kappa}\|x^0\|_{\Upsilon^2 R}^2. \tag{194}$$

Therefore From (113) and Lemma 5.7 we have that

$$4\|x^0\|_{\Upsilon^2 R}^2 \leq \frac{4\left(\widetilde{P}^0 - f(x^0)\right)}{\widetilde{\kappa}}$$

$$\overset{(113)}{\leq} \frac{4\left(f(x^0) + \frac{5}{M}d_0^\top L^{-1} d_0 - \underline{f}\right)}{\widetilde{\kappa}} := \frac{4\widetilde{C}_1}{\widetilde{\kappa}}.$$

Combining the above two relations leads to

$$\frac{1}{T_r} \sum_{r=0}^{T_r} \left( \left\| \frac{\sum_{i=1}^{M} \nabla f_i(x_i^r)}{M} \right\|^2 + \|\Sigma F x^r\|^2 \right)$$

$$\leq 128 \left( \sum_{i=1}^{M} \beta_i^2 + 3 + \frac{1}{32\widetilde{\kappa}} \right) \widetilde{C}_1 / T_r.$$

This completes the proof. **Q.E.D.**

# References

[1] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," in *Advances in Neural Information Processing Systems*, 2017.

[2] P. A. Forero, A. Cano, and G. B. Giannakis, "Distributed clustering using wireless sensor networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 707–724, Aug 2011.

[3] T.-H. C. H.-T. Wai and A. Scaglione, "A consensus-based decentralized algorithm for non-convex optimization with application to dictionary learning," in *the Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015.

[4] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[5] A. Nedic and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 601–615, 2015.

[6] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2014.

[7] D. Jakovetić, J. M. Moura, and J. Xavier, "Linear convergence rate of a class of distributed augmented lagrangian algorithms," *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 922–936, 2015.

[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[9] I. Schizas, G. Mateos, and G. Giannakis, "Distributed LMS for consensus-based in-network adaptive processing,," *IEEE Transactions on Signal Processing*, vol. 57, no. 6, pp. 2365 – 2382, 2009.

[10] P. Bianchi and J. Jakubowicz, "Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 2, pp. 391–405, 2013.

[11] M. Zhu and S. Martínez, "An approximate dual subgradient algorithm for distributed non-convex constrained optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 6, pp. 1534–1539, June 2013.

[12] M. Hong, Z.-Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM Journal On Optimization*, vol. 26, no. 1, pp. 337–364, 2016.

[13] P. Di Lorenzo and G. Scutari, "Next: In-network nonconvex optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 120–136, 2016.

[14] D. Hajinezhad and M. Hong, "Perturbed proximal primal dual algorithm for nonconvex nonsmooth optimization," *Mathematical Programming*, vol. 176, no. 1-2, pp. 207–245, July 2019.

[15] M. Hong, D. Hajinezhad, and M.-M. Zhao, "Prox-PDA: The proximal primal-dual algorithm for fast distributed nonconvex optimization and learning over networks," in *the Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[16] D. Hajinezhad, M. Hong, and A. Garcia, "Zone: Zeroth order nonconvex multi-agent optimization over networks," *IEEE Transactions on Automatic Control*, 2019.

[17] A. Daneshmand, G. Scutari, and F. Facchinei, "Distributed dictionary learning," in *Proceedings of the Asilomar Conference on Signals, Systems, and Computers*, Nov. 6–9, 2016.

[18] A. Daneshmand, Y. Sun, G. Scutari, and F. Facchinei, "Distributed dictionary learning over networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, March 5-9 2017.

[19] J. Zeng and W. Yin, "On nonconvex decentralized gradient descent," *IEEE Transactions on Signal Processing*, vol. 66, no. 11, pp. 2834–2848, June 2018.

[20] Z. Jiang, A. Balu, C. Hegde, and S. Sarkar, "Collaborative deep learning in fixed topology networks," in *Advances in Neural Information Processing Systems*, 2017.

[21] S. Vlaski and A. H. Sayed, "Distributed learning in non-convex environments–part i: Agreement at a linear rate," *arXiv preprint arXiv:1907.01848*, 2019.

[22] ——, "Distributed learning in non-convex environments–part ii: Polynomial escape from saddle-points," *arXiv preprint arXiv:1907.01849*, 2019.

[23] B. Swenson, S. Kar, H. V. Poor, and J. Moura, "Annealing for distributed global optimization," *arXiv preprint arXiv:1903.07258*, 2019.

[24] M. Hong, J. D. Lee, and M. Razaviyayn, "Gradient primal-dual algorithm converges to second-order stationary solutions for nonconvex distributed optimization," in *the Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

[25] A. Daneshmand, G. Scutari, and V. Kungurtsev, "Second-order guarantees of gradient algorithms over networks," in *Proceedings of the 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2018.

[26] B. Swenson, R. Murray, H. V. Poor, and S. Kar, "Distributed gradient descent: Nonconvergence to saddle points and the stable-manifold theorem," *arXiv preprint arXiv:1908.02747*, 2019.

[27] C. Duenner, A. Lucchi, M. Gargiani, A. Bian, T. Hofmann, and M. Jaggi, "A distributed second-order algorithm you can trust," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[28] C.-H. Fang, S. B. Kylasa, F. Roosta-Khorasani, M. W. Mahoney, and A. Grama, "Distributed second-order convex optimization," *arXiv preprint arXiv:1807.07132*, 2018.

[29] Y. Nesterov, "Smooth minimization of nonsmooth functions," *Mathematical Programming*, vol. 103, pp. 127–152, 2005.

[30] ——, "A method of solving a convex programming problem with convergence rate $o(1/k^2)$," *Soviet Mathematics Doklady*, vol. 27, pp. 372–376, 1983.

[31] A. Nemirovsky and D. Yudin, "Problem complexity and method efficiency in optimization," in *Interscience Series in Discrete Mathematics*. Wiley, 1983.

[32] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer, 2004.

[33] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imgaging Science*, vol. 2, no. 1, pp. 183 – 202, 2009.

[34] Y. Ouyang, Y. Chen, G. Lan, and J. E. Pasiliao, "An accelerated linearized alternating direction method of multipliers," *SIAM Journal on Imaging Sciences*, vol. 8, no. 1, pp. 644–681, 2015.

[35] P. Tseng, "On accelerated proximal gradient methods for convex-concave optimization," 2008, preprint.

[36] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Fast distributed gradient methods," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1131–1146, May 2014.

[37] K. Scaman, F. Bach, S. Bubeck, Y. Lee, and L. Massoulié, "Optimal algorithms for smooth and strongly convex distributed optimization in networks," *arXiv preprint arXiv:1702.08704*, 2017.

[38] C. Uribe, S. Lee, A. Gasnikov, and A. Nedić, "Optimal algorithms for distributed optimization," *arXiv preprint arXiv:1712.00232*, 2017.

[39] K. Scaman, F. Bach, S. Bubeck, L. Massoulié, and Y. T. Lee, "Optimal algorithms for non-smooth distributed optimization in networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 2740–2749.

[40] C. Cartis, N. Gould, and P. Toint, "On the complexity of steepest descent, newton's and regularized newton's methods for nonconvex unconstrained optimization problems," *SIAM journal on optimization*, vol. 20, no. 6, pp. 2833–2852, 2010.

[41] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford, "Lower bounds for finding stationary points i," *Mathematical Programming*, Jun 2019.

[42] Y. Tian, Y. Sun, B. Du, and G. Scutari, "Asy-sonata: Achieving geometric convergence for distributed asynchronous optimization," *arXiv preprint arXiv:1803.10359*, 2018.

[43] A. Daneshmand, Y. Sun, and G. Scutari, "Convergence rate of distributed convex and nonconvex optimization methods with gradient tracking," 2018, purdue University, Tech. Rep.

[44] X. Fu, K. Huang, N. Sidiropolous, A. M.-S. So, and M. Hong, "Scalable and optimal generalized canonical correlation analysis via alternating optimization," 2016, submitted to NIPS 2016.

[45] F. R. K. Chung, *Spectral Graph Theory*. The American Mathematical Society, 1997.

[46] S. Butler, *Algebraic aspects of the normalized Laplacian*. Cham: Springer International Publishing, 2016, pp. 295–315.

[47] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, March 2012.

[48] H. Uzawa, "Iterative methods in concave programming," in *Studies in Linear and Nonlinear Programming*. Stanford University Press, 1958, p. 154165.

[49] A. Nedić and A. Ozdaglar, "Subgradient methods for saddle-point problems," *Journal of optimization theory and applications*, vol. 142, no. 1, pp. 205–228, 2009.

[50] R. T. Rockafellar, "Augmented lagrangians and applications of the proximal point algorithm in convex programming," *Mathematics of operations research*, vol. 1, no. 2, pp. 97–116, 1976.

[51] S. J.Wright, "Implementing proximal point methods for linear programming," *Journal of Optimization Theory and Applications*, vol. 65, no. 3, pp. 531–554, Jun 1990.

[52] D. Tian, H. Mansour, A. Knyazev, and A. Vetro, "Chebyshev and conjugate gradient filters for graph image denoising," in *Proceedings of the IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, 2014.

[53] A. Gadde, S. K. Narang, and A. Ortega, "Bilateral filter: Graph spectral interpretation and extensions," in *Proceedings of the IEEE International Conference on Image Processing*.

[54] V. S. Ryaben'kii and S. V. Tsynkov, *A Theoretical Introduction to Numerical Analysis*. CRC Press, 2007.

[55] A. A. Samarskij and E. S. Nikolaev, *Numerical Methods for Grid Equations Volume II Iterative Methods*. Springer, 1989.

[56] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[57] A. Antoniadis, I. Gijbels, and M. Nikolova, "Penalized likelihood regression for generalized linear models with non-quadratic penalties," *Annals of the Institute of Statistical Mathematics*, vol. 63, no. 3, pp. 585–615, 2011.

[58] T. Tatarenko and B. Touri, "Non-convex distributed optimization," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3744–3757, 2017.

[59] A. Nedić, A. Olshevsky, A. Ozdaglar, and J. N. Tsitsiklis, "On distributed averaging algorithms and quantization effects," *IEEE Transactions on Automatic Control*, vol. 54, no. 11, pp. 2506–2517, 2009.

[60] T.-H. Chang, M. Hong, and X. Wang, "Multi-agent distributed optimization via inexact consensus ADMM," *IEEE Transactions on Signal Processing*, vol. 63, no. 2, pp. 482–497, Jan 2015.