Distributed Optimization: Algorithms and Convergence Rates

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Dušan Jakovetić

Dipl. Ing., School of Electrical Engineering, University of Belgrade

Carnegie Mellon University, Pittsburgh, PA

Instituto Superior Técnico (IST), Technical University of Lisbon

May 2013

To my parents, Mirjana and Milisav.

Acknowledgement

Dušan Jakovetić thanks the thesis committee members, co-advisor José M. F. Moura (Carnegie Mellon University, Pittsburgh, PA, USA), co-advisor João Xavier (Instituto Superior Técnico, Technical University of Lisbon, Lisbon, Portugal), Thomas Marzetta (Communications and Signal Processing Research Group, Bell Laboratories, Alcatel-Lucent, Murray Hill, NJ, USA), Mário Figueiredo (Instituto Superior Técnico, Technical University of Lisbon, Lisbon, Portugal), Soummya Kar (Carnegie Mellon University, Pittsburgh, PA, USA), Rohit Negi (Carnegie Mellon University, Pittsburgh, PA, USA), and João Pedro Gomes (Instituto Superior Técnico, Technical University of Lisbon, Lisbon, Portugal).

Dušan Jakovetić also acknowledges support from the Fundação de Ciência e Tecnologia (FCT), Portugal.

Abstract

This thesis develops and analyzes distributed algorithms for convex optimization in networks, when nodes cooperatively minimize the sum of their locally known costs subject to a global variable of common interest. This setup encompasses very relevant applications in networked systems, including distributed estimation and source localization in sensor networks, and distributed learning. Generally, existing literature offers two types of distributed algorithms to solve the above problem: 1) distributed (consensus-based) gradient methods; and 2) distributed augmented Lagrangian methods; but both types present several limitations. 1) Distributed gradient-like methods have slow practical convergence rate; further, they are usually studied for very general, non-differentiable costs, and the possibilities for speed-ups on more structured functions are not sufficiently explored. 2) Distributed augmented Lagrangian methods generally show good performance in practice, but there is a limited understanding of their convergence rates, specially how the rates depend on the underlying network.

This thesis contributes to both classes of algorithms in several ways. We propose a new class of fast distributed gradient algorithms that are Nesterov-like. We achieve this by exploiting the structure of convex, differentiable costs with Lipschitz continuous and bounded gradients. We establish their fast convergence rates in terms of the number of per-node communications, per-node gradient evaluations, and the network spectral gap. Furthermore, we show that current distributed gradient methods cannot achieve the rates of our methods under the same function classes. Our distributed Nesterov-like gradient algorithms achieve guaranteed rates for both static and random networks, including the scenario with intermittently failing links or randomized communication protocols. With respect to distributed augmented Lagrangian methods, we consider both deterministic and randomized distributed methods, subsuming known methods but also introducing novel algorithms. Assuming twice continuously differentiable costs with a bounded Hessian, we establish global linear convergence rates, in terms of the number of per-node communications, and, unlike most of the existing work, in terms of the network spectral gap. We illustrate our methods with several applications in sensor networks and distributed learning.

Contents

Chapter 1

Introduction

1.1 Motivation and thesis objective

Motivated by applications in sensor, multi-robot, and cognitive networks, as well as in distributed learning, we consider synthesis and analysis of distributed optimization algorithms. We study the problem where each node i in a connected network acquires data D_i to infer a vector quantity $x^\star \in \mathbb{R}^d$. Node i's own data D_i give only partial knowledge on $x^\star$, but the quantity $x^\star$ can be successfully reconstructed based on all nodes' data. More formally, each node i has a local convex cost function $f_i(x) = f_i(x; D_i)$ of the variable x (parameterized by data D_i), known only to node i. The goal is for each node to find $x^\star$ that solves the unconstrained problem:

$$\text{minimize} \ \sum_{i=1}^{N} f_i(x) =: f(x). \tag{1.1}$$

(See also Figure 1.1 for an illustration of the model with a generic network of $N = 6$ nodes.) Problem (1.1) is highly relevant as it encompasses, among many other network applications, the following: 1) distributed inference (detection and estimation) [3] and distributed source localization in sensor networks [4]; 2) spectrum sensing [5] and resource allocation [6] in cognitive radio networks; 3) emulation of swarms in biological networks [7]; and 4) distributed Lasso [8], linear classification [9], and other learning problems. Section 1.5 details several applications in networked systems that we consider throughout the thesis.

The main objectives of this thesis are to: 1) develop distributed iterative algorithms whereby each node i obtains a solution to (1.1); and 2) establish convergence and convergence rate guarantees of these methods, in the presence of inter-neighbor communication failures and communication asynchrony. At iteration k, nodes exchangetheir current solution estimate $x_i(k)$ (and, possibly, an additional low-overhead quantity) only with their immediate neighbors in the network. Nodes do not exchange the raw data D_i, either due to

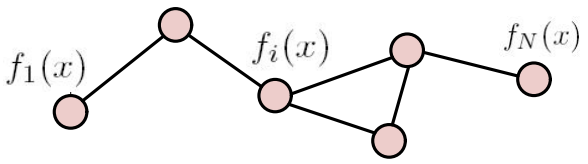

Figure 1.1: An example of a connected network with $N = 6$ nodes. Each node i has a local convex cost function $f_i(x)$.

extremely high dimension of D_i, as may arise with massive distributed learning problems, or due to privacy constraints.

The remainder of the introductory Chapter is organized as follows. Section 1.2 states our main contributions. Section 1.3 reviews existing literature and contrasts it with this thesis. Section 1.4 outlines novel technical tools that this thesis developed and that may find use in other contexts. Finally, we finish the Chapter in Section 1.5 with a list of motivating engineering applications, explored throughout the thesis.

1.2 Thesis contributions

To solve (1.1), in general, existing literature proposes two types of algorithms: 1) distributed (consensus-based) gradient-type algorithms; and 2) distributed augmented Lagrangian (AL) dual algorithms. This thesis contributes to both types of methods.

Distributed gradient-type methods. With respect to distributed gradient-type methods, our main contributions include the following.

- We develop novel distributed gradient methods that converge significantly faster than existing distributed gradient methods;
- We establish global convergence rates of our methods in terms of the cost function parameters (e.g., Lipschitz constant of the gradient) and the underlying network parameters; remarkably, acceleration techniques guarantee convergence rates (in expectation) even on random networks;
- We show that existing distributed gradient methods cannot achieve the rates of our methods under equal network and cost functions conditions.

Distributed augmented Lagrangian methods. With respect to distributed augmented Lagrangian methods, our main contributions include the following.

- We develop novel distributed AL algorithms that operate with asynchronous inter-node communication;

- We establish global linear convergence rates of a wide class of distributed AL methods under the convex twice continuously differentiable costs with bounded Hessian, in terms of the overall per-node communications at any stage of the algorithm.

We now detail each of our contributions, namely, on distributed nesterov-like gradient methods, distributed consensus and averaging, and distributed AL methods.

Chapters 2–4: Distributed Nesterov-like gradient methods

We propose distributed Nesterov-like gradient methods, and we establish their convergence rate guarantees for both static and random networks. Random networks account for random packet dropouts with wireless sensor networks, as well as random inter-node communication protocols, like gossip [10]. In Chapters 2 and 3, we achieve this on the class $\mathcal{F}$ of convex, differentiable costs f_i's that have Lipschitz continuous and bounded gradients. Chapter 4 establishes the convergence rates for alternative function classes, hence further broadening the applications that our methods can handle. We now detail specific contributions in Chapters 2, 3, and 4.

Chapter 2: Static networks

We propose two distributed gradient methods based on a Nesterov's centralized fast gradient method [11]. We assume the f_i's in (1.1) are in class $\mathcal{F}$. We refer to our first method as Distributed Nesterov-Gradient (D–NG for short). With D–NG, each node i in the network updates its solution estimate $x_i(k)$ at time k by: 1) weight-averaging its own and its neighbors' estimates (we denote by W_{ij} the weights and by O_i the neighborhood of node i, including i); 2) performing a negative local gradient step with steps size α_k with respect to f_i; and 3) updating an auxiliary variable $y_i(k)$ to accelerate convergence via a distributed version of a Nesterov-like step [11]:

$$x_i(k+1) = \sum_{j \in O_i} W_{ij}\, y_j(k) - \alpha_k \nabla f_i(y_i(k)) \tag{1.2}$$

$$y_i(k+1) = x_i(k+1) + \frac{k}{k+3}\left(x_i(k+1) - x_i(k)\right). \tag{1.3}$$

Iterations (1.2)–(1.3) are computationally simple; the price paid with respect to the standard distributed gradient method [2] is negligible. However, we show that our method achieves, at each node i, the convergence

rates in the optimality gap at the cost function $\frac{1}{N}\left(f(x_i) - f^\star\right)$:

$$O\left(\frac{1}{(1-\mu)^{1+\xi}}\frac{\log k}{k}\right) \text{ and } O\left(\frac{1}{(1-\mu)^{1+\xi}}\frac{\log \mathcal{K}}{\mathcal{K}}\right), \tag{1.4}$$

in the number of per-node communications $\mathcal{K}$ and per-node gradient evaluations k. In (1.4), the quantity $(1-\mu) \in (0,1]$ is the network's spectral gap[1], and $\xi > 0$ is arbitrarily small.

We refer to our second method, whose details are in Chapter 3, as Distributed Nesterov gradient with Consensus iterations (D–NC for short.) Under the class $\mathcal{F}$, D–NC achieves convergence rates:

$$O\left(\frac{1}{(1-\mu)^2}\frac{1}{\mathcal{K}^{2-\xi}}\right) \text{ and } O\left(\frac{1}{k^2}\right). \tag{1.5}$$

(See also Table 1.1, rows 1 and 2.)[2] Both distributed gradient methods D–NG and D–NC show significant gains over existing, standard distributed gradient methods [2]. We show that existing methods cannot perform better than $\Omega\left(\frac{1}{k^{2/3}}\right)$ and $\Omega\left(\frac{1}{\mathcal{K}^{2/3}}\right)$.[3] In other words, our methods achieve significantly faster rates than [2] on the class $\mathcal{F}$, both in terms of k and $\mathcal{K}$, while maintaining algorithm simplicity.

Chapter 3: Random Networks

We modify our D–NG and D–NC methods to handle *random networks*. We refer to the modified methods as mD–NG and mD–NC, respectively. We model the network by a sequence of independent, identically distributed matrices $W(k)$, drawn from the set of symmetric, stochastic[4] matrices with positive diagonals. We assume that the underlying network is connected on average[5]. We establish with mD–NG and mD–NC methods the rates in terms of the *expected* normalized optimality gap $\frac{1}{N}\left(\mathbb{E}\left[f(x_i)\right] - f^\star\right)$, at arbitrary node i, as a function of k, $\mathcal{K}$, the number of nodes N, and the quantity $1-\overline{\mu}$ that is a generalization of the spectral gap $1-\mu$ for random networks.[6] The rates of mD–NG and mD–NC for random networks are shown in Table 1.1., rows 3 and 4. We can see that, in expectation, mD–NG and mD–NC achieve the same rates in k and $\mathcal{K}$ that D–NG and D–NC achieve on static networks. Hence, remarkably, acceleration ideas of Nesterov apply also to random networks and allow for much faster algorithms than offered by the existing literature.

[1]The quantity μ is the modulus of the second largest (in modulus) eigenvalue of the $N \times N$ matrix W that collects the weights W_{ij}'a.

[2]The rates' constants shown in the Table 1.1 are for the optimized step-sizes; for details and constants under generic step-sizes, we refer to Chapters 2 and 3.

[3]For two positive sequences a_k and b_k, notation $b_k = O(a_k)$ means that $\limsup_{k\to\infty} \frac{b_k}{a_k} < \infty$; $b_k = \Omega(a_k)$ means that $\liminf_{k\to\infty} \frac{b_k}{a_k} > 0$; and $b_k = \Theta(a_k)$ if $b_k = O(a_k)$ and $b_k = \Omega(a_k)$.

[4]Stochastic means that the rows of the matrix sum to one and all its entries are nonnegative.

[5]This means that the graph that supports the nonzero entries of the expected matrix $\mathbb{E}\left[W(k)\right]$ is connected.

[6]More precisely, $\overline{\mu}^2$ is the second largest eigenvalue of the second moment $\mathbb{E}\left[W(k)^2\right]$.

	Gradient evaluations	Communications
D–NG (Static):	$O\left(\frac{1}{(1-\mu)^{1+\xi}}\frac{\log k}{k}\right)$	$O\left(\frac{1}{(1-\mu)^{1+\xi}}\frac{\log \mathcal{K}}{\mathcal{K}}\right)$
D–NC (Static):	$O\left(\frac{1}{k^2}\right)$	$O\left(\frac{1}{(1-\mu)^2}\frac{1}{\mathcal{K}^{2-\xi}}\right)$
mD–NG (Random):	$O\left(\frac{N}{(1-\overline{\mu})^{4/3}}\frac{\log k}{k}\right)$	$O\left(\frac{N}{(1-\overline{\mu})^{4/3}}\frac{\log \mathcal{K}}{\mathcal{K}}\right)$
mD–NC (Random):	$O\left(\frac{1}{k^2}\right)$	$O\left(\frac{N^{\xi}}{(1-\overline{\mu})^2}\frac{1}{\mathcal{K}^{2-\xi}}\right)$

Table 1.1: Convergence rates in the normalized cost function optimality gaps $\frac{1}{N}(f(x_i) - f^\star)$ at arbitrary node i for the proposed distributed Nesterov-like gradient methods: D–NG, D–NC, mD–NG, and mD–NC. The cost functions f_i's belong to the class $\mathcal{F}$ of convex, differentiable, costs, with Lipschitz continuous and bounded gradients.

As explained above, existing distributed gradient methods cannot achieve better rates than $\Omega(1/k^{2/3})$ and $\Omega(1/\mathcal{K}^{2/3})$ even on static networks.

Chapter 4: Alternative function classes

We establish convergence and convergence rate guarantees for our distributed gradient methods under problem classes different than $\mathcal{F}$. In particular, we do not explicitly require that the gradients be bounded, and we allow for constrained optimization, where each node i has the same closed, convex constraint set $\mathcal{X}$. Thus, we enlarge the class of applications for our distributed gradient methods with provable rate guarantees. In particular, we show the following: 1) We establish rates $O(\log k/k)$ and $O(\log \mathcal{K}/\mathcal{K})$ for D–NG and unconstrained problems, when the bounded gradients assumption is replaced with a certain growth condition; 2) We establish convergence to ϵ-solution neighborhood (in terms of the cost function) of D–NG under a constant step-size after $O(1/\epsilon)$ per-node communications and per-node gradient evaluations, for constrained optimization with a compact constrain set; and 3) We establish rates $O(1/k^2)$ and $O(1/\mathcal{K}^{2-\xi})$ of the mD–NC method for constrained optimization with compact constraints. Further details on the assumed problem setups can be found in Chapter 4.

Chapter 5: Weight Optimization for Consensus in Random Networks

Performance of distributed consensus-based gradient algorithms, like our D–NG and D–NC and their modifications mD–NG and mD–NC, as well as [2] and [12], significantly depend on the underlying averaging (consensus) dynamics – see the weighed averaging in (1.2). Consensus dynamics significantly depends on the weights W_{ij}'s in (1.2) that nodes assign to their neighbors. We address the problem of *optimal weight*

design such that consensus dynamics are the fastest possible, allowing for random networks with spatially correlated link failures. Our weight design is clearly of direct interest for consensus and distributed averaging, but also allows for faster convergence of distributed optimization methods, as we demonstrate by simulation in Chapter 5. Our weight design is applicable to: 1) networks with correlated random link failures (see, e.g., [13] and 2) networks with randomized algorithms (see, e.g, [10, 1]). We address the weight design problem for both symmetric and asymmetric random links.

With symmetric random links, we use as optimization criterion the mean squared consensus convergence rate that equals $\overline{\mu}^2$. We express express the rate as a function of the link occurence probabilities, their correlations, and the weights. We prove that $\overline{\mu}^2$ is a convex, nonsmooth function of the weights. This enables global optimization of the weights for arbitrary link occurrence probabilities and arbitrary link correlation structures. We provide insights into weight design with a simple example of complete random network that admits closed form solution for the optimal weights and convergence rate and show how the optimal weights depend on the number of nodes, the link occurence probabilities, and their correlations. Finally, we extend our results to the case of asymmetric random links, adopting as an optimization criterion the mean squared deviation (from the current average state) rate, and show that this metric is a convex function of the weights. Simulation examples demonstrate the gains with our weight design compared with existing weight assignment choices, both in distributed averaging and in distributed optimization.

Chapters 6 and 7: Distributed augmented Lagrangian (AL) methods

A popular approach to solve (1.1) is through the AL dual; we briefly outline it here to help us state our contributions. (Derivations can be found in Chapter 6.) Each node i updates its solution estimate $x_i(k)$ (primal variable), and a dual variable $\eta_i(k)$, by:

$$(x_1(k+1), ..., x_N(k+1)) = \arg\min_{(x_1,...,x_N) \in \mathbb{R}^{dN}} L_a (x_1, ..., x_N; \eta_1(k), ..., \eta_N(k)) \tag{1.6}$$

$$\eta_i(k+1) = \eta_i(k) + \alpha \sum_{j \in O_i} W_{ij} \, (x_i(k+1) - x_j(k+1)) , \tag{1.7}$$

where $\alpha > 0$ is the (dual) step-size, $L_a : \mathbb{R}^{dN} \times \mathbb{R}^{dN} \to \mathbb{R}$, is the AL function:

$$L_a(x_1, \ldots, x_N; \eta_1, ..., \eta_N) = \sum_{i=1}^{N} f_i(x_i) + \sum_{i=1}^{N} \eta_i^\top x_i + \frac{\rho}{2} \sum_{\{i,j\} \in E, \, i<j} W_{ij} \, \|x_i - x_j\|^2, \tag{1.8}$$

$\rho \geq 0$ is the AL penalty parameter, and E is the edge set. Dual updates (1.7) are performed in a distributed way, as each node i needs only the primal variables $x_j(k+1)$ from its immediate neighbors in the network.

When $\rho = 0$, the primal update (1.6) decouples as well, and node i solves for $x_i(k+1)$ locally (without inter-neighbor communications.) When $\rho > 0$ (which is advantageous in many sistuations), the quadratic coupling term in (1.8) induces the need for an iterative solution of (1.6) that involves the inter-neighbor communications.

Our main contributions with respect to (1.6)–(1.7) are two-fold: 1) We design efficient novel algorithms to iteratively solve (1.6); and 2) We establish convergence rates for the overall scheme (1.6)–(1.7), in terms of the *overall* per-node communications $\mathcal{K}$ at any algorithm stage.

We now highlight our specific contributions.

Chapter 6: Distributed augmented Lagrangian algorithms for generic costs

We propose a randomized distributed AL method, termed Augmented Lagrangian algorithm with Gossip communication (AL–G), that: 1) handles very general, nondifferentiable costs f_i's; 2) allows a private constraint set $\mathcal{X}_i$ at each node i; and 3) utilizes asynchronous, unidirectional, gossip communication. This contrasts with existing distributed AL methods that handle only static networks and synchronous communication, while the AL–G algorithm handles random networks and uses gossip communication. In distinction with existing distributed gradient algorithms that essentially handle only symmetric link failures, AL–G handles asymmetric link failures.

The AL–G method operates in two time scales. Nodes update their dual variable synchronously via a step of type (1.7),[7] at a slow time scale. To solve (1.6), nodes update their primal variables via a *novel* nonlinear Gauss-Seidel method. The method uses asynchronous, gossip operations, at a fast time scale, as follows. At a time, one node is selected at random; upon selection, it updates its primal variable x_i, and transmits it to a randomly selected neighbor. We further propose two variants of AL–G, the AL–BG (augmented Lagrangian with broadcast gossiping) and AL–MG (augmented Lagrangian with multi-neighbor gossiping) methods.

We prove convergence of the inner primal algorithms (1.6), when the number of inner iterations goes to infinity. This establishes convergence of the nonlinear Gauss-Seidel method with *random* order of minimizations, while existing literature previously showed convergence only under the *cyclic* or the *essentially cyclic* rules, [14, 15]. We illustrate the performance of AL–G with two simulation examples, l_1–regularized logistic regression for classification and cooperative spectrum sensing for cognitive radios.

[7] The step is similar but not the same; we refer to Chapter 6 for details.

Chapter 7: Distributed augmented Lagrangian algorithms for costs with bounded Hessian

We assume a restricted class of functions with respect to Chapter 6, namely convex, twice differentiable f_i's with a bounded Hessian. We establish globally linear convergence rates for a class of methods of type (1.6)–(1.7) in the overall per-node communications $\mathcal{K}$. Furthermore, we give explicit dependence of the convergence rate on the network spectral gap $1-\mu$.

Specifically, we study a class of deterministic and randomized methods of type (1.6)–(1.7). Both deterministic and randomized methods update the dual variables via (1.7). With the deterministic variants, step (1.6) is done via multiple inner iterations of either: 1) the nonlinear Jacobi (NJ) method on $L(\cdot;\,\eta(k))$; or 2) the gradient descent on $L(\cdot;\,\eta(k))$. With both cases, one inner iteration corresponds to one per-node broadcast communication to all neighbors. With the randomized methods, step (1.6) is done either via multiple inner iterations of: 1) a randomized nonlinear Gauss-Seidel (NGS) method on $L(\cdot;\,\eta(k))$ (this variant is precisely the AL–BG method from the previous Chapter); or 2) a randomized coordinate gradient descent on $L(\cdot;\,\eta(k))$. Hence, we consider the total of four algorithm variants: 1) deterministic NJ; 2) deterministic gradient; 3) randomized NGS (this is AL–BG in Chapter 6); and 4) randomized gradient. With all variants, we establish linear convergence rates in the total number of elapsed per-node communications $\mathcal{K}(k)$ after k outer iterations, assuming that nodes know beforehand a Hessian lower bound $h_{\min}$, a Hessian upper bound $h_{\max}$, and (a lower bound) on the network spectral gap $1-\mu$. The distance to the solution $x^\star$ of (1.1), at any node i and any outer iteration k decays as:[8]

$$\|x_i(k)-x^\star\| = O\left(\mathcal{R}^{\mathcal{K}(k)}\right), \tag{1.9}$$

with the communication rate $\mathcal{R}\in[0,1)$ (the smaller it is, the better) shown in Table 1.2. The quantity $\gamma := h_{\max}/h_{\min}$ is the condition number of the f_i's. Our results, in contradistinction with the literature, establish how the convergence rate depends on the underlying network (spectral gap $1-\mu$.) For example, for the deterministic NJ method (row 1 in Table 1.2), the rate is jointly negatively affected by the condition number γ and the network "connectivity," measured by $1-\mu$. For a poorly connected chain network of N nodes, $1-\mu=\Theta(1/N^2)$, and hence the rate is $1-\Omega\left(\frac{1}{(1+\gamma)N^2}\right)$. In contrast, for well-connected expander networks, $1-\mu=\Omega(1)$, i.e., it stays bounded away from one, and so the rate essentially does not deteriorate with the increase of N.

[8] With randomized methods, the result is in terms of the expected distance $\mathbb{E}\left[\|x_i-x^\star\|\right]$ and in terms of the expected overall communications $\mathbb{E}\left[\mathcal{K}(k)\right]$, see Chapter 7 for details.

	Convergence Rate $\mathcal{R}$
Deterministic, NJ:	$1-\Omega\left(\frac{1-\mu}{1+\gamma}\right)$
Deterministic, Gradient:	$1-\Omega\left(\frac{1-\mu}{1+\gamma}\,\frac{\log\left(1+\frac{1}{1+\gamma}\right)}{\log(1+\gamma)+\log\left((1-\mu)^{-1}\right)}\right)$
Randomized, NGS:	$1-\Omega\left(\frac{1-\mu}{1+\gamma}\,\frac{1}{\log(1+\gamma)+\log\left((1-\mu)^{-1}\right)}\right)$
Randomized, Gradient:	$1-\Omega\left(\frac{1-\mu}{(1+\gamma)^2}\,\frac{1}{\log(1+\gamma)+\log\left((1-\mu)^{-1}\right)}\right)$

Table 1.2: Convergence rates $\mathcal{R}$ in (1.9) for the four variants of (1.6)–(1.7).

1.3 Review of the literature

We briefly review the literature to contrast ours with existing work. We consider: (1) distributed gradient methods; and (2) distributed AL methods.

Distributed gradient methods

Distributed gradient methods have been studied in the past [16, 17], with main focus on parallel architectures (star networks) and on distributing the optimization variable's components across different processors (for generic networks). More recent literature proposes and analyzes: 1) diffusion algorithms, e.g., [18, 19]; 2) primal (sub)gradient methods [2, 20, 21, 22, 23, 24, 25, 26]; 3) dual averaging methods [12, 4, 27]; and 4) primal-dual (sub)gradient methods [28]. These methods are usually studied under a broad class of convex f_i's (possibly non-differentiable) with bounded gradients. With respect to convergence rate, [12] establishes the convergence rates $O\left(\frac{\log k}{k^{1/2}}\right)$ and $O\left(\frac{\log \mathcal{K}}{\mathcal{K}^{1/2}}\right)$ for the algorithm presented in [12]. Recent references [29, 30] that appeared after our initial works on Nesterov-like gradient methods develop and analyze accelerated proximal methods for time varying networks that resemble D–NC, but have no parallels with our D–NG methods. Their rates are worse than ours in terms of the number of nodes N. (We contrast in detail our work with [29, 30] in Chapter 2.)

In summary, we identify the following limitations of the existing work on distributed gradient-like methods:

1. The methods show a slow (theoretical and practical) convergence rates; see, e.g., our simulation examples in Chapter 2.

2. Theoretical studies are available only for very wide classes of non-differentiable costs, where it is not possible to achieve fast convergence rates. In particular, no faster rates than $O(1/\sqrt{k})$ are possible,

even in a centralized setting. The literature does not sufficiently explore the relevant scenario of more restricted classes of functions, where in a centralized setting faster rates are known to be achievable. For example, with differentiable costs that have Lipschitz continuous gradients, centralized gradient methods can achieve rates $O(1/k^2)$.

3. Acceleration techniques by Nesterov and others for gradient methods have not been sufficiently explored.

This thesis and [29, 30] respond to the above limitations. In particular, we achieve the following: 1) We propose fast distributed gradient methods D–NG, D–NC, and their modifications mD–NG and mD–NC. Our methods converge faster than existing distributed gradient methods, both in theory and in simulation. 2) We study distributed gradient methods under the class $\mathcal{F}$ when the costs f_i's are differentiable, and have Lipschitz continuous and bounded gradients. This setup allows for convergence rates $O(1/k^2)$ in a centralized setting. Our distributed D–NC method achieves rates $O(1/k^2)$ and $O(1/\mathcal{K}^{2-\xi})$ and is hence close to the best achievable (centralized) rates. 3) We show that the acceleration techniques due to Nesterov allow for speed-ups in distributed optimization as well, remarkably, even on random networks.

Distributed augmented Lagrangian methods

In a classical, centralized setting, augmented Lagrangian (AL) and alternating direction method of multipliers (ADMM) methods have been studied for a long time; e.g., references [31, 32, 33] show *locally* linear or superlinear convergence rates of AL methods. Recently, there has been a strong renewed interest and progress in the convergence rate analysis of the classical AL and ADMM methods. References [34, 35] show that the ADMM method converges *globally* linearly for certain cost function classes.

Reference [36] analyzes the AL methods under more general costs than ours and is not concerned with the distributed optimization problem (1.1). The work [36] is related to ours in the sense that it analyzes the AL methods when the primal problems are solved inexactly, but, under their setup, the AL methods converge to a solution neighborhood. By contrast, in our setup studied in Chapter 7, distributed AL methods converge linearly to the exact solution, in spite of the inexact solutions of the (inner) primal problems.

Lagrangian dual methods for *parallel* computation (star configuration) have been studied in the past [37]. More recently, the literature considers *generic* networks, e.g., [8, 38, 39, 40] for AL methods, and [41] for ordinary dual methods. References [40, 8, 5] propose different versions of the distributed alternating direction method of multipliers methods (ADMM). They have been particularly extensively studied in the signal processing literature, demonstrating their effectiveness in various applications, e.g., [42, 43, 39, 8, 40].

These works do not study the convergence rates of their proposed methods. The results in [34, 35] may imply linear convergence of the D-Lasso and related methods in, e.g., [8, 5]. Hence, the results in [34, 35] may imply linear convergence for a *sub-class* of methods that we consider in this thesis. Our analysis is technically different from the analysis in [34, 35].

Reference [44] considers the distributed optimization problem (1.1) over generic networks as we do here, under a wider class of functions than what we study. The reference shows $O\left(1/\mathcal{K}\right)$ rate of convergence in the number of per-node communications for a distributed ADMM method. Hence, with respect to our work, [44] studies a wider class of problems but establishes much slower rates. Reference [45] considers both resource allocation problems and (1.1) and develops accelerated dual gradient methods. For (1.1), this reference gives the methods' local rates as $1-\Omega\left(\sqrt{\frac{\lambda_{\min}(AA^{\top})}{\gamma\,\lambda_{\max}(AA^{\top})}}\right)$, where A is the edge-node incidence matrix and $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ denote the minimal and maximal eigenvalues, respectively. An important distinction is that [45] considers ordinary dual problems, with the AL parameter $\rho=0$; in contrast, we consider both cases $\rho=0$ and $\rho>0$.

In summary, we identify the following limitations of the literature on *distributed* AL and ADMM methods:

1. The literature offers a limited understanding on how global convergence rates depend on the underlying network parameters, e.g., spectral gap.

2. There is a limited study of the convergence rates of these algorithms in an asynchronous operation, or in the presence of random link failures.

We respond to the above literature limitations as follows: 1) We establish globally linear convergence rates for a wide class of distributed AL algorithms; 2) We explicitly characterize the rates in terms of the network spectral gap; and 3) We establish linear convergence rates (in expectation) for distributed AL methods that utilize asynchronous communication.

Portions of this thesis have been published in journal papers [46, 38], submitted to journals [9, 47], and are to be submitted to a journal in [48]; and published in conference proceedings [49, 50].

During the course of this thesis, we also published journal [3, 51, 52] and conference [53, 54, 55, 56, 57, 58] papers.

1.4 Technical tools developed in the thesis

This thesis develops technical tools that are, to the authors' best knowledge, not available in the literature, and may find use in further developments of distributed optimization or other contexts. These tools are detailed below.

Stability of random time-varying systems

In Chapters 2 and 3, we analyze the following time-varying random systems that arise with distributed methods that employ Nesterov's acceleration techniques:

$$\widetilde{z}(k) = \left(\begin{bmatrix} (1+\beta_{k-1}) & -\beta_{k-1} \\ 1 & 0 \end{bmatrix} \otimes (W(k) - J) \right) \widetilde{z}(k-1) + \frac{1}{k} u(k-1), \;\; k = 1, 2, ..., \qquad (1.10)$$

where $\otimes$ is the Kronecker product; $\widetilde{z}(k)$ is the $2N \times 1$ system state; $J = \frac{1}{N} 1 1^\top$; $\beta_k = k/(k+3)$; $W(k)$ is a random, symmetric, stochastic matrix; and $u(k)$ is an almost surely bounded random vector. System (1.10) involves $2N$-dimensional states, it is random, time varying, and more complex than the systems that arise with standard distributed gradient methods, e.g., [2, 12]. We show that:

$$\mathbb{E}\left[\|\widetilde{z}(k)\|^2\right] = O\left(\frac{N^4}{(1-\bar{\mu})^4 \, k^2}\right),$$

hence establishing the mean square convergence rate to zero. We refer to Chapters 2 and 3 for further details.

Inexact projected Nesterov gradient method

We establish optimality gap bounds for the inexact projected and non-projected (centralized) Nesterov gradient method [11], in the presence of gradient inexactness and projection inexactness. This contributes to existing studies of such methods [59, 60]. Reference [60] considers only gradient inexactness and a different Nesterov's method. Reference [59] considers both gradient and projections (in fact, inexact proximal steps). However, it establishes bounds in terms of the *norms* of the gradient inexactness ϵ_k. In contrast, we establish (in this context, refined) bounds in terms of the *squared norms* of gradient inexactness ϵ_k^2. While the bounds that involve ϵ_k are sufficient to study the methods of type D–NC and [29, 30], they do not allow for meaningful bounds for D–NG type methods; for such methods, one needs to work with the bounds that involve ϵ_k^2. We refer to Chapters 2 and 3 for further details.

Lower bounds on the performance of distributed gradient methods

We establish lower bounds on the performance of distributed gradient like methods. Our techniques construct "hard" problem instances. Generally, they account for the fact that, with distributed consensus-based methods like D–NG, D–NC, and [2, 12], the local nodes' gradients at a global solution $x^\star$ of (1.1) are non-zero. In particular, we show that [2, 12] cannot achieve worst-case rates better than $\Omega(1/k^{2/3})$, as well as that all three methods D–NG, D–NC, and [2] are arbitrarily slow in the worst case when the functions do not have uniformly bounded gradients. We refer to Chapters 2 and 3 for further details.

Analysis of augmented Lagrangian methods with a fixed number of inner iterations

We analyze augmented Lagrangian methods of type (1.6)–(1.7) with a *fixed* number of inner iterations so that problems (1.6) are solved inexactly. When the f_i's are twice continuously differentiable with a bounded Hessian, we establish globally linear convergence to the exact solution in the primal domain, in terms of the overall communications (overall number of the inner iterations.) This contrasts with the literature that usually 1) uses an increasing number τ_k of the inner iterations at the outer iteration k; or 2) establishes only convergence to a solution neighborhood. We refer to Chapter 7 for further details.

1.5 Motivating applications

We give several examples of problem (1.1) from existing literature that are relevant in engineering applications; we study each of these examples in subsequent Chapters.

Example 1: Consensus

We explain consensus in the context of sensor networks, but many other contexts, e.g., social networks, are possible [61]. Let N nodes (sensors) be deployed in a field; each sensor acquires a scalar measurement d_i, e.g., temperature at its location. The goal is for each sensor to compute the average temperature in the field: $\frac{1}{N}\sum_{i=1}^{N} d_i$. Consensus can be cast as (1.1) by setting $f_i(x) = \frac{1}{2}(x - d_i)^2$.

Example 2: Distributed learning: linear classifier

For concreteness, we focus on linear classification, but other distributed learning problems fit naturally (1.1). Training data (e.g., data about patients, as illustrated in [62]) is distributed across nodes in the network (different hospitals); each node has N_s data samples, $\{a_{ij}, b_{ij}\}_{j=1}^{N_s}$, where $a_{ij} \in \mathbb{R}^m$ is a feature vector

(patient signature – blood pressure, etc) and $b_{ij} \in \{-1, +1\}$ is the class label of the vector a_{ij} (patient healthy or ill). For the purpose of future feature vector classifications, each node wants to learn the linear classifier $a \mapsto \text{sign}\left(a^\top x' + x''\right)$, i.e., to determine a vector $x' \in \mathbb{R}^m$ and a scalar $x'' \in \mathbb{R}$, based on all nodes' data samples, that makes the best classification in a certain sense. Specifically, we seek $x' \in \mathbb{R}^m$ and $x'' \in \mathbb{R}$ that minimize a convex surrogate loss with respect to $x = ((x')^\top, x'')^\top$:

$$\text{minimize} \quad \sum_{i=1}^{N} \sum_{j=1}^{N_s} \phi\left(-b_{ij}(a_{ij}^\top x' + x'')\right) + \nu\, R(x') \ . \tag{1.11}$$

Here $\phi(z)$ is a surrogate loss function, e.g., logistic, exponential, or hinge loss [63], $\nu > 0$ is a parameter, and $R(x')$ is the regularization, e.g., l_1 norm or quadratic. Problem (1.11) fits (1.1), with $f_i(x) := \sum_{j=1}^{N_s} \phi\left(-b_{ij}(a_{ij}^\top x' + x'')\right) + \frac{\nu}{N} R(x')$.

Example 3: Acoustic source localization in sensor networks

A sensor network instruments the environment where an acoustic source is positioned at an unknown location $\theta \in \mathbb{R}^2$, e.g., [64]. The source emits a signal isotropically. Each node (sensor) i measures the received signal strength:

$$y_i = \frac{A}{\|\theta - r_i\|^\beta} + \zeta_i.$$

Here $r_i \in \mathbb{R}^2$ is node i's location, known to node i, $A > 0$ and $\beta > 0$ are constants known to all nodes, and ζ_i is zero-mean additive noise. The goal is for each node to estimate the source's position θ. A straightforward approach is to find the nonlinear least squares estimate $\overline{\theta} = x^\star$ by minimizing the following cost function (of the variable x):

$$\text{minimize} \quad \sum_{i=1}^{N} \left(y_i - \frac{A}{\|x - r_i\|^\beta}\right)^2 \ . \tag{1.12}$$

Problem (4.81) is nonconvex and hence difficult; still, it is possible to efficiently obtain an estimator $\widehat{\theta}$ based on the data y_i, $i = 1, ..., N$, by solving the following *convex* problem:

$$\text{minimize} \quad \tfrac{1}{2} \sum_{i=1}^{N} \text{dist}^2\left(x, C_i\right) \ , \tag{1.13}$$

where C_i is the disk $C_i = \left\{x \in \mathbb{R}^2 : \|x - r_i\| \leq \left(\frac{A}{y_i}\right)^{1/\beta}\right\}$, and $\text{dist}(x, C) = \inf_{y \in C} \|x - y\|$ is the distance from x to the set C. In words, (4.82) finds a point $\widehat{\theta}$ that has the minimal total squared distance from disks C_i, $i = 1, ..., N$. Problem (4.82) fits our framework (1.1) with $f_i(x) = \frac{1}{2}\text{dist}^2\left(x, C_i\right)$.

Example 4: Spectrum sensing for cognitive radio networks

Consider N secondary users (CRs) connected by a generic network. The CRs sense the power spectral density (PSD) to reconstruct the PSD map of primary users (PUs), i.e., the CRs want to determine at what physical locations the PUs are present, and what frequencies they use; this example is studied in [5, 8]. The model assumes N_p potential locations (a grid) of PUs; each "potential" PU p has a power spectral density (PSD) expressed as:

$$\Phi_p(\omega) = \sum_{b=1}^{N_b} \theta_{bp} \Psi_b(\omega),$$

where ω is the frequency, $\Psi_b(\omega)$ is rectangle over interval b, and θ_{bp} is a coefficient that says how much PSD is generated by the pth (potential) PU in the frequency range b. The PSD at CR i is modeled as a superposition of all potential PU's PSDs:

$$\Phi_i(\omega) = \sum_{p=1}^{N_p} g_{ip} \Phi_p(\omega) = \sum_{p=1}^{N_p} g_{ip} \sum_{b=1}^{N_b} \theta_{bp} \Psi_b(\omega), \tag{1.14}$$

where g_{ip} is the channel gain between PU p and CR i. Denote by θ the vector that stacks all the θ_{bp}'s. Each CR collects samples at frequencies ω_l, $l = 1, ..., L$, modeled as:

$$y_{i,l} = \Phi_i(\omega_l) + \zeta_{i,l} = h_{i,l}(\theta) + \zeta_{i,l},$$

where $\zeta_{i,l}$ is a zero-mean additive noise, and $h_{i,l}(\theta)$ is a linear function of θ. Reference [5] assumes that most of the coefficients θ_{bp} are zero, i.e., the vector θ is sparse. Hence, the spectrum sensing problem of determining the vector $x^\star$ – an estimate of θ – is:

$$\text{minimize} \quad \textstyle\sum_{i=1}^{N} \sum_{l=1}^{L} (h_{i,l}(x) - y_{i,l})^2 + \nu \|x\|_1, \tag{1.15}$$

where $\|x\|_1$ is the l_1-norm that sums the absolute values of the entries of the vector. In framework (1.1), we now have $f_i(x) = \sum_{l=1}^{L} (h_{i,l}(x) - y_{i,l})^2 + \frac{\nu}{N} \|x\|_1$.

We consider Example 1 is Chapter 5; example 2 in Chapters 2, 6, and 7; example 3 in Chapter 4; and example 4 in Chapter 6.

Comment on organization. The chapters are designed such that it is possible to read them independently. Each chapter carefully explains the problem considered, the assumptions underlying the chapter, and the new algorithms and corresponding results. The price paid is some repetition in problem formulation and

set-up.

Chapter 2

Distributed Nesterov-like Gradient Methods: Static Networks

2.1 Introduction

In this Chapter, we propose distributed Nesterov-like gradient methods for static networks that solve (1.1). We consider the class $\mathcal{F}$ of convex, differentiable costs f_i's, with Lipschitz continuous and bounded gradients. Existing work [2, 12] usually assumes a wider class, where the f_i's are (possibly) non-differentiable and convex, and: 1) for unconstrained minimization, the f_i's have bounded gradients, while 2) for constrained minimization, they are Lipschitz continuous over the constraint set. It is well established in centralized optimization, [11], that one expects faster convergence rates on classes of more structured functions; e.g., for convex, non-smooth functions, the best achievable rate for centralized (sub)gradient methods is $O(1/\sqrt{k})$, while, for convex functions with Lipschitz continuous gradient, the best rate is $O(1/k^2)$, achieved, e.g., by the Nesterov gradient method [11]. Here k is the number of iterations, i.e., the number of gradient evaluations.

Contributions. Building from the centralized Nesterov gradient method, we develop for the class $\mathcal{F}$ two distributed gradient methods and prove their convergence rates, in terms of the number of per-node communications $\mathcal{K}$, the per-node gradient evaluations k, and the network topology. Our first method, the Distributed Nesterov Gradient (D–NG), achieves convergence rate $O\left(\frac{1}{(1-\mu(W))^{3+\xi}}\frac{\log k}{k}\right)$, where $\xi > 0$ is an arbitrarily small quantity, when the nodes have no global knowledge of the parameters underlying the optimization problem and the network: L and G the f_i's gradient's Lispchitz constant and the gradient bound, respectively, $1-\mu(W)$ the spectral gap, and R a bound on the distance to a solution. When L and $\mu(W)$ are known

by all, the Distributed Nesterov Gradient method with optimized step-size achieves $O\left(\frac{1}{(1-\mu(W))^{1+\xi}}\frac{\log k}{k}\right)$.

Our second method, Distributed Nesterov gradient with Consensus iterations (D–NC), assumes global knowledge or at least upper bounds on $\mu(W)$ and L. It achieves convergence rate $O\left(\frac{1}{(1-\mu(W))^2}\frac{1}{\mathcal{K}^{2-\xi}}\right)$ in the number of communications per node $\mathcal{K}$, and $O\left(\frac{1}{k^2}\right)$ in the number of gradient evaluations. Further, we establish that, for the class $\mathcal{F}$, both our methods (achieving at least $O(\log k/k)$) are strictly better than the distributed (sub)gradient method [2] and the distributed dual averaging in [12], even when these algorithms are restricted to functions in $\mathcal{F}$. We show analytically that [2] cannot be better than $\Omega\left(1/k^{2/3}\right)$ and $\Omega\left(1/\mathcal{K}^{2/3}\right)$ (see Subsection 2.7.1), and by simulation examples that [2] and [12] perform similarly.

Distributed versus centralized Nesterov gradient methods. The centralized Nesterov gradient method does not require bounded gradients – an assumption that we make for our distributed methods. We prove here that if we drop the bounded gradients assumption, the convergence rates that we establish do not hold for either of our algorithms. (It may be possible to *replace* the bounded gradients assumption with a weaker requirement.) In fact, the worst case convergence rates of D–NG and D–NC become arbitrarily slow. (See Subsection 2.7.2 for details.) This important result illustrates a distinction between the allowed function classes by the centralized and distributed methods. The result is not specific to our accelerated methods; it can be shown that the standard distributed gradient method in [2] is also arbitrarily slow when the assumption of bounded gradients is dropped (while convexity and Lipschitz continuous gradient hold), see Appendix A.

Remark. The bounded gradients Assumption cannot be relaxed, but we show that it can be replaced with certain different setups, with both D–NG and D–NC methods. We consider these alternative setups in Chapter 4.

Remark. We comment on references [29] and [30] (see also Subsection 2.7.1 and Appendix A). They develop accelerated proximal methods for time varying networks that resemble D–NC. The methods in [29] and [30] use only one consensus algorithm per outer iteration k, while we use two with D–NC. Adapting the results in [29, 30] to our framework, it can be shown that the optimality gap bounds in [29, 30] expressed in terms of $N, 1-\mu(W)$, and $\mathcal{K}$ have the same or worse (depending on the variant of their methods) dependence on $\mathcal{K}$ and $\mu(W)$ than the one we show for D–NC, and a worse dependence on N. (See Subsection 2.7.1 and Appendix A.)

Chapter organization. The next paragraph introduces notation. Section 2.2 describes the network and optimization models that we assume. Section 2.3 presents our algorithms, the distributed Nesterov gradient and the distributed Nesterov gradient with consensus iterations, D–NG and D–NC for short. Section 2.4 explains the framework of the (centralized) inexact Nesterov gradient method; we use this framework to establish the convergence rate results for D–NG and D–NC. Sections 2.5 and 2.6 prove convergence rate

results for the algorithms D–NG and D–NC, respectively. Section 2.7 compares our algorithms D–NG and D–NC with existing distributed gradient type methods, discusses the algorithms' implementation, and discusses the need for our Assumptions. Section 2.9 provides simulation examples. Finally, we conclude in Section 2.10. Proofs of certain lengthy arguments are relegated to Appendix A.

Notation. We index by a subscript i a (possibly vector) quantity assigned to node i; e.g., $x_i(k)$ is node i's estimate at iteration k. Further, we denote by: $\mathbb{R}^d$ the d-dimensional real coordinate space; $\mathbf{j}$ the imaginary unit ($\mathbf{j}^2 = -1$); A_{lm} or $[A]_{lm}$ the entry in the l-th row and m-th column of matrix A; $a^{(l)}$ the l-th entry of vector a; $(\cdot)^\top$ the transpose and $(\cdot)^H$ the conjugate transpose; I, 0, $\mathbf{1}$, and e_i, respectively, the identity matrix, the zero matrix, the column vector with unit entries, and the i-th column of I; $\oplus$ and $\otimes$ the direct sum and Kronecker product of matrices, respectively; $\|\cdot\|_l$ the vector (respectively, matrix) l-norm of its vector (respectively, matrix) argument; $\|\cdot\| = \|\cdot\|_2$ the Euclidean (respectively, spectral) norm of its vector (respectively, matrix) argument ($\|\cdot\|$ also denotes the modulus of a scalar); $\lambda_i(\cdot)$ the i-th smallest *in modulus* eigenvalue; $A \succeq 0$ means that a Hermitian matrix A is positive semi-definite; $\lceil a \rceil$ the smallest integer not smaller than a real scalar a; $\nabla\phi(x)$ and $\nabla^2\phi(x)$ the gradient and Hessian at x of a twice differentiable function $\phi : \mathbb{R}^d \to \mathbb{R}$, $d \geq 1$. For two positive sequences a_k and b_k, the following is the standard notation: $b_k = O(a_k)$ if $\limsup_{k\to\infty} \frac{b_k}{a_k} < \infty$; $b_k = \Omega(a_k)$ if $\liminf_{k\to\infty} \frac{b_k}{a_k} > 0$; and $b_k = \Theta(a_k)$ if $b_k = O(a_k)$ and $b_k = \Omega(a_k)$.

Part of the material in Chapter 2 has been submitted for publication in [9].

2.2 Model and preliminaries

In this Section, we present our model and preliminaries. Specifically, Subsection 2.2.1 introduces the network and optimization models that we assume, subsection 2.2.2 reviews the consensus algorithm, and Subsection 2.2.1 reviews the centralized Nesterov gradient method.

2.2.1 Model

Network model

We consider a (sparse) network $\mathcal{N}$ of N nodes (sensors, processors, agents,) each communicating only locally, i.e., with a subset of the remaining nodes. The communication pattern is captured by the graph $\mathcal{G} = (\mathcal{N}, E)$, where $E \subset \mathcal{N} \times \mathcal{N}$ is the set of links. The graph $\mathcal{G}$ is connected, undirected and simple (no self/multiple links.)

Weight matrix. We associate to the graph $\mathcal{G}$ a symmetric, doubly stochastic (rows and columns sum to one and all the entries are non-negative), $N \times N$ weight matrix W, with, for $i \neq j$, $W_{ij} > 0$ if and only if, $\{i, j\} \in E$, and $W_{ii} = 1 - \sum_{j \neq i} W_{ij}$. Denote by $\widetilde{W} = W - J$, where $J := \frac{1}{N}\mathbf{1}\mathbf{1}^\top$ is the ideal consensus matrix. We let $\widetilde{W} = Q\widetilde{\Lambda}Q^\top$, where $\widetilde{\Lambda}$ is the diagonal matrix with $\widetilde{\Lambda}_{ii} = \lambda_i(\widetilde{W})$, and $Q = [q_1, ..., q_N]$ is the matrix of the eigenvectors of $\widetilde{W}$. With D–NC, we impose Assumption 2.1 (a) below; with D–NG, we require both Assumptions 2.1 (a) and (b).

Assumption 2.1 (Weight matrix) We assume that (a) $\mu(W) < 1$; and (b) $W \succeq \kappa I$, where $\kappa < 1$ is an arbitrarily small positive quantity.

Note that Assumption 2.1 (a) can be fulfilled only by a connected network. Assumption 2.1 (a) is standard and is also needed with the existing algorithms in [2, 12]. For a connected network, nodes can assign the weights W and fulfill Assumption 2.1 (a), e.g., through the Metropolis weights [65]; to set the Metropolis weights, each node needs to know its own degree and its neighbors' degrees. Assumption 2.1 (b) required by D–NG is not common in the literature. We discuss the impact of Assumption 2.1 (b) in Subsection 2.7.1.

Distributed optimization model

The nodes solve the unconstrained problem:

$$\text{minimize} \ \sum_{i=1}^{N} f_i(x) =: f(x). \tag{2.1}$$

The function $f_i : \mathbb{R}^d \to \mathbb{R}$ is known only to node i. We impose Assumptions 2.2 and 2.3.

Assumption 2.2 (Solvability and Lipschitz continuity of the derivative) (a) There exists a solution $x^\star \in \mathbb{R}^d$ with $f(x^\star) = \inf_{x \in \mathbb{R}^d} f(x) =: f^\star$.

(b) $\forall i$, f_i is convex, differentiable, with Lipschitz continuous derivative with constant $L \in [0, \infty)$:

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|, \quad \forall x, y \in \mathbb{R}^d.$$

Assumption 2.3 (Bounded gradients) $\exists G \in [0, \infty)$ such that, $\forall i$, $\|\nabla f_i(x)\| \leq G$, $\forall x \in \mathbb{R}^d$.

Examples of f_i's that satisfy Assumptions 2.2–2.3 include the logistic and Huber losses (See Section 2.9), or the "fair" loss in robust statistics, $\phi : \mathbb{R} \mapsto \mathbb{R}$, $\phi(x) = b_0^2 \left(\frac{|x|}{b_0} - \log\left(1 + \frac{|x|}{b_0}\right)\right)$, where b_0

is a positive parameter, e.g., [66]. Assumption 2.2 is precisely the assumption required by [11] in the convergence analysis of the (centralized) Nesterov gradient method. With respect to the centralized Nesterov gradient method [11], we additionally require bounded gradients as given by Assumption 2.3. We explain the need for Assumption 2.3 in Subsection 2.7.2. We refer to Chapter 4 for different optimization models.

2.2.2 Consensus algorithm

We review the standard consensus algorithm that serves as a building block for our D–NG and D–NC methods, as well as other distributed methods in the literature, e.g., [2, 12]. Suppose that each node i in a network acquires a scalar measurement $z_i(0)$. The goal for each node i is to obtain the global average of the measurements $\overline{z}(0) := \frac{1}{N}\sum_{i=1}^{N} z_i(0)$, by communicating only with its immediate neighbors in the network. This can be accomplished by the standard consensus algorithm, whereby nodes iteratively update their state $z_i(k)$ as:

$$z_i(k) = \sum_{j \in O_i} W_{ij}\, z_j(k-1), \;\; k = 1, 2, ... \tag{2.2}$$

Here, W_{ij} are the averaging weights (the entries of W), and O_i is the neighborhood set of node i (including i). Operation of consensus is as follows. At iteration k, node i transmits its state $z_i(k-1)$ to all neighbors $j \in O_i - \{i\}$, as well as receives the states $z_j(k-1)$, for $j \in O_i - \{i\}$. Upon reception, each node i makes the weighted average as in (2.2). Introducing the network-wide state vector $z(k) = (z_1(k), ..., z_N(k))^\top$, consensus is rewritten in compact form as:

$$z(k) = W\, z(k-1), \;\; k = 1, 2, ... \tag{2.3}$$

2.2.3 Centralized Nesterov gradient method

We briefly review the fast (centralized) gradient method proposed by Nesterov in [11]. Consider a differentiable convex function $f : \mathbb{R}^d \to \mathbb{R}$ that has Lipschitz continuous gradient with constant L. The goal is to find $x^\star \in \mathbb{R}^d = \arg\min_{x \in \mathbb{R}^d} f(x)$. (We assume that such a $x^\star$ exists.) The fast centralized gradient method [11] updates the solution estimate $\overline{x}(k)$ and an auxiliary variable $\overline{y}(k)$ as follows:[1]

$$\overline{x}(k) \;\; = \;\; \overline{x}(k-1) - \alpha\, \nabla f(\overline{y}(k-1)) \tag{2.4}$$

$$\overline{y}(k) \;\; = \;\; \overline{x}(k) + \beta_{k-1}\, (\, \overline{x}(k) - \overline{x}(k-1)\,), \;\; k = 1, 2, ... \tag{2.5}$$

[1]For convenience, we denote the iterates here by $\overline{x}(k)$ and $\overline{y}(k)$. We will show later that the global averages, also denoted by $\overline{x}(k), \overline{y}(k)$, of the nodes estimates with our distributed methods evolve according to an inexact version of (2.4)–(2.5).

with the initialization $\overline{x}(0) = \overline{y}(0) \in \mathbb{R}^d$. The constant step size $\alpha \leq 1/L$ and $\beta_k = k/(k+3)$, for $k = 0, 1, ...$ Compared with the standard gradient method:

$$\overline{x}(k) = \overline{x}(k-1) - \alpha \nabla f(\overline{x}(k-1)), \;\; k = 1, 2, ...,$$

the Nesterov gradient method introduces an auxiliary variable $\overline{y}(k)$ and an inexpensive update (2.5). With this small overhead per iteration k, the Nesterov gradient method significantly increases the convergence rate in the cost function optimality gap $f(\overline{x}(k)) - f(x^\star)$, from $O(1/k)$ to $O(1/k^2)$ [11].[2] There seems to be little explanation in the literature about the geometric intuition of the Nesterov gradient method. See, e.g., ([67], Chapter 5.3), for relations of the Nesterov gradient method with the earlier proposed heavy ball method by Polyak and the conjugate gradient method; see also ([68], Chapter 3.2). Finally, we remark that there exist other variants of fast gradient methods in [69, 70]; see also for a method for composite, nondifferentiable optimization [71], and [72].

2.3 Distributed Nesterov based algorithms

We now consider our two proposed algorithms. Subsection 2.3.1 presents algorithm D–NG, while subsection 2.3.2 presents algorithm D–NC.

2.3.1 Distributed Nesterov gradient algorithm (D–NG)

Algorithm D–NG generates the sequence $(x_i(k), y_i(k))$, $k = 0, 1, 2, ...$, at each node i, where $y_i(k)$ is an auxiliary variable. D–NG is initialized by $x_i(0) = y_i(0) \in \mathbb{R}^d$, for all i. The update at node i and iteration $k = 1, 2, ...$ is:

$$x_i(k) = \sum_{j \in O_i} W_{ij}\, y_j(k-1) - \alpha_{k-1} \nabla f_i(y_i(k-1)) \tag{2.6}$$

$$y_i(k) = x_i(k) + \beta_{k-1} \left(x_i(k) - x_i(k-1) \right). \tag{2.7}$$

Here, the step-size α_k and the sequence β_k are:

$$\alpha_k = \frac{c}{k+1}, \quad c > 0; \quad \beta_k = \frac{k}{k+3}, \quad k = 0, 1, ... \tag{2.8}$$

[2]The convergence rate $O(1/k^2)$ with the Nesterov gradient method holds on the class of convex, differentiable costs, with Lipschitz continuous gradient of constant L.

With algorithm (2.6)–(2.7), each node i, at each iteration k, performs the following: 1) broadcasts its variable $y_i(k-1)$ to all its neighbors $j \in O_i$; 2) receives $y_j(k-1)$ from all its neighbors $j \in O_i$; 3) updates $x_i(k)$ by weight-averaging its own $y_i(k-1)$ and its neighbors variables $y_j(k-1)$, and performs a negative gradient step with respect to f_i; and 4) updates $y_i(k)$ via the inexpensive update in (2.7). To avoid notation explosion in the analysis further ahead, we assume throughout the chapter, with both D–NG and D–NC, equal initial estimates $x_i(0) = y_i(0) = x_j(0) = y_j(0)$ for all i, j; e.g., nodes can set them to zero.

We adopt the sequence β_k as in the centralized fast gradient method by Nesterov [11]; see also [72, 73]. With the centralized Nesterov gradient, $\alpha_k = \alpha$ is constant along the iterations. However, under a constant step-size, algorithm (2.6)–(2.7) does not converge to the exact solution, but only to a solution neighborhood. More precisely, in general, $f(x_i(k))$ does not converge to $f^\star$ (See [49] for details.) We force $f(x_i(k))$ to converge to $f^\star$ with (2.6)–(2.7) by adopting a diminishing step-size α_k, as in (2.8). The constant $c > 0$ in (2.8) can be arbitrary (See also ahead Theorem 2.8.)

Vector form. Let $x(k) = (x_1(k)^\top, x_2(k)^\top, ..., x_N(k)^\top)^\top$, $y(k) = (y_1(k)^\top, y_2(k)^\top, ..., y_N(k)^\top)^\top$, and introduce $F : \mathbb{R}^{Nd} \to \mathbb{R}$ as: $F(x) = F(x_1, x_2, ..., x_N) = f_1(x_1) + f_2(x_2) + ..., f_N(x_N)$. Then, given initialization $x(0) = y(0)$, D–NG in vector form is:

$$x(k) = (W \otimes I)y(k-1) - \alpha_{k-1}\nabla F(y(k-1)) \tag{2.9}$$

$$y(k) = x(k) + \beta_{k-1}\left(x(k) - x(k-1)\right), \quad k = 1, 2, ..., \tag{2.10}$$

where the identity matrix is of size d – the dimension of the optimization variable in (2.1).

2.3.2 Algorithm D–NC

Algorithm D–NC uses a *constant step-size* $\alpha \leq 1/(2L)$ and operates in two time scales. In the outer (slow time scale) iterations k, each node i updates its solution estimate $x_i(k)$, and updates an auxiliary variable $y_i(k)$ (as with the D–NG);[3] in the inner iterations s, nodes perform two rounds of consensus with the number of inner iterations given in (2.11) and (2.12) below, respectively. D–NC is Summarized in Algorithm 1.

The number of inner consensus iterations in (2.11) increases as $\log k$ and depends on the underlying network through $\mu(W)$. Note an important difference between D–NC and D–NG. D–NC uses explicitly a number of consensus steps at each k. In contrast, D–NG does not explicitly use multi-step consensus at each k; consensus occurs implicitly, similarly to [2, 12].

Vector form. Using the same compact notation for $x(k)$, $y(k)$, and $\nabla F(y(k))$ as with D–NG, D–NC in

[3] To avoid notation explosion, we use the same letters to denote the iterates of D–NG and D–NC.

Algorithm 1 Algorithm D–NC

1: Initialization: Node i sets: $x_i(0) = y_i(0) \in \mathbb{R}^d$; and $k = 1$.

2: Node i calculates: $x_i^{(a)}(k) = y_i(k-1) - \alpha \nabla f_i(y_i(k-1))$.

3: (First consensus) Nodes run average consensus initialized by $x_i^{(c)}(s=0,k) = x_i^{(a)}(k)$:

$$x_i^{(c)}(s,k) = \sum_{j \in O_i} W_{ij} x_j^{(c)}(s-1,k), \;\; s = 1,2,...,\tau_x(k), \qquad \tau_x(k) = \left\lceil \frac{2\log k}{-\log \mu(W)} \right\rceil, \tag{2.11}$$

and set $x_i(k) := x_i^{(c)}(s = \tau_x(k), k)$.

4: Node i calculates $y_i^{(a)}(k) = x_i(k) + \beta_{k-1}\left(x_i(k) - x_i(k-1)\right)$.

5: (Second consensus) Nodes run average consensus initialized by $y_i^{(c)}(s=0,k) = y_i^{(a)}(k)$:

$$y_i^{(c)}(s,k) = \sum_{j \in O_i} W_{ij} y_j^{(c)}(s-1,k), \;\; s = 1,2,...,\tau_y(k), \qquad \tau_y(k) = \left\lceil \frac{\log 3}{-\log \mu(W)} + \frac{2\log k}{-\log \mu(W)} \right\rceil, \tag{2.12}$$

and set $y_i(k) := y_i^{(c)}(s = \tau_y(k), k)$.

6: Set $k \mapsto k+1$ and go to step 2.

vector form is:

$$x(k) \;\; = \;\; (W \otimes I)^{\tau_x(k)} \left[\; y(k-1) - \alpha \, \nabla F(y(k-1)) \; \right] \tag{2.13}$$

$$y(k) \;\; = \;\; (W \otimes I)^{\tau_y(k)} \left[\; x(k) + \beta_{k-1}(x(k) - x(k-1)) \; \right]. \tag{2.14}$$

The power $(W \otimes I)^{\tau_x(k)}$ in (2.13) corresponds to the first consensus in (2.11), and the power $(W \otimes I)^{\tau_y(k)}$ in (2.14) corresponds to the second consensus in (2.12). The connection between D–NC and the (centralized) Nesterov gradient method becomes clearer in Subsection 2.4.2. The matrix powers (2.13)–(2.14) are implemented in a distributed way through multiple iterative steps – they require respectively $\tau_x(k)$ and $\tau_y(k)$ iterative (distributed) consensus steps. This is clear from the representation in Algorithm 1.

2.4 Intermediate results: Inexact Nesterov gradient method

We will analyze the convergence rates of D–NG and D–NC by considering the evolution of the global averages $\overline{x}(k) := \frac{1}{N}\sum_{i=1}^N x_i(k)$ and $\overline{y}(k) := \frac{1}{N}\sum_{i=1}^N y_i(k)$. We will show that, with both distributed methods, the evolution of $\overline{x}(k)$ and $\overline{y}(k)$ can be studied through the framework of the inexact (centralized) Nesterov gradient method, essentially like the one in [60]. Subsection 2.4.1 introduces this framework and gives the relation for the progress in one iteration. Subsection 2.4.2 then demonstrates that we can cast our algorithms D–NG and D–NC in this framework.

2.4.1 Inexact Nesterov gradient method

We next introduce the definition of a (pointwise) inexact first order oracle.

Definition 2.4 (Pointwise inexact first order oracle) Consider a function $f : \mathbb{R}^d \to \mathbb{R}$ that is convex and has Lipschitz continuous gradient with constant L_f. We say that a pair $\left(\widehat{f}_y, \widehat{g}_y\right) \in \mathbb{R} \times \mathbb{R}^d$ is a (L_y, δ_y) inexact oracle of f at point y if:

$$\widehat{f}_y + \widehat{g}_y^\top (x - y) \leq f(x) \leq \widehat{f}_y + \widehat{g}_y^\top (x - y) + \frac{L_y}{2}\|x - y\|^2 + \delta_y, \quad \forall x \in \mathbb{R}^d. \tag{2.15}$$

For any $y \in \mathbb{R}^d$, the pair $(f(y), \nabla f(y))$ satisfies Definition 4.11 with $(L_y = L_f, \delta_y = 0)$. If $\left(\widehat{f}_y, \widehat{g}_y\right)$ is a (L_y, δ_y) inexact oracle at y, then it is also a $\left(L'_y, \delta_y\right)$ inexact oracle at y, with $L'_y \geq L_y$.

Remark. The prefix pointwise in Definition 4.11 emphasizes that we are concerned with finding $\left(\widehat{f}_y, \widehat{g}_y\right)$ that satisfy (4.49) with (L_y, δ_y) *at a fixed point* y. This differs from the conventional definition (Definition 1) in [60]. Throughout, we always refer to the inexact oracle in the sense of Definition 4.11 here and drop the prefix pointwise.

Inexact Nesterov gradient method

Lemma 2.5 gives the progress in one iteration of the inexact (centralized) Nesterov gradient method for the unconstrained minimization of f. Consider a point $(\overline{x}(k-1), \overline{y}(k-1)) \in \mathbb{R}^d \times \mathbb{R}^d$, for some fixed $k = 1, 2, ...$ Let $\left(\widehat{f}_{k-1}, \widehat{g}_{k-1}\right)$ be a (L_{k-1}, δ_{k-1}) inexact oracle of the function f at point $\overline{y}(k-1)$ and:[4]

$$\overline{x}(k) = \overline{y}(k-1) - \frac{1}{L_{k-1}}\widehat{g}_{k-1}, \quad \overline{y}(k) = \overline{x}(k) + \beta_{k-1}\left(\overline{x}(k) - \overline{x}(k-1)\right). \tag{2.16}$$

Lemma 2.5 (Progress per iteration) Consider the update rule (4.51) for some $k = 1, 2, ...$ Then:

$$\begin{aligned} & (k+1)^2 \left(f(\overline{x}(k)) - f(x^\bullet)\right) + 2L_{k-1}\|\overline{v}(k) - x^\bullet\|^2 \\ \leq \; & (k^2 - 1)\left(f(\overline{x}(k-1)) - f(x^\bullet)\right) + 2L_{k-1}\|\overline{v}(k-1) - x^\bullet\|^2 + (k+1)^2\delta_{k-1}, \end{aligned} \tag{2.17}$$

for any $x^\bullet \in \mathbb{R}^d$, where $\gamma_k = 2/(k+2)$ and $\overline{v}(k) = \frac{\overline{y}(k) - (1-\gamma_k)\overline{x}(k)}{\gamma_k}$.

Lemma 2.5 is similar to [[60], Theorem 5], although [60] considers a different accelerated Nesterov method. It is intuitive: the progress per iteration is the same as with the exact Nesterov gradient algorithm, except that

[4]For convenience, we denote the iterates in (4.51) by $\overline{x}(k), \overline{y}(k)$ – the same notation as we use for the global averages with D–NG and D–NC, as we later apply Lemma 2.5 to these global averages.

it is deteriorated by the "gradient direction inexactness" ($(k+1)^2\delta_{k-1}$). The proof follows the arguments of [60] and [73, 11, 72] and is in Appendix A.

2.4.2 Algorithms D–NG and D–NC in the inexact oracle framework

We now cast algorithms D–NG and D–NC in the inexact oracle framework.

Algorithm D–NG

Recall the global averages $\overline{x}(k) := \frac{1}{N}\sum_{i=1}^N x_i(k)$ and $\overline{y}(k) := \frac{1}{N}\sum_{i=1}^N y_i(k)$, and define:

$$\widehat{f}_k = \sum_{i=1}^N \Big\{ f_i(y_i(k)) + \nabla f_i(y_i(k))^\top (\overline{y}(k) - y_i(k)) \Big\}, \quad \widehat{g}_k = \sum_{i=1}^N \nabla f_i(y_i(k)). \tag{2.18}$$

Multiplying (2.9)–(2.10) from the left by $(1/N)(\mathbf{1}^\top \otimes I)$, using $(\mathbf{1}^\top \otimes I)(W \otimes I) = \mathbf{1}^\top \otimes I$, letting $L'_{k-1} := \frac{N}{\alpha_{k-1}}$, and using $\widehat{g}_k$ in (2.18), we obtain that $\overline{x}(k)$, $\overline{y}(k)$ evolve according to:

$$\overline{x}(k) = \overline{y}(k-1) - \frac{1}{L'_{k-1}}\widehat{g}_{k-1}, \qquad \overline{y}(k) = \overline{x}(k) + \beta_{k-1}\left(\overline{x}(k) - \overline{x}(k-1)\right), \tag{2.19}$$

The following Lemma shows how we can analyze convergence of (2.19) in the inexact oracle framework. Define $\widetilde{y}_i(k) := y_i(k) - \overline{y}(k)$ and $\widetilde{y}(k) := (\widetilde{y}_1(k)^\top, ..., \widetilde{y}_N(k))^\top$. Define analogously $\widetilde{x}_i(k)$ and $\widetilde{x}(k)$. We refer to $\widetilde{x}(k)$ and $\widetilde{y}(k)$ as the disagreement vectors, as they indicate how mutually apart the estimates of different nodes are.

Lemma 2.6 Let Assumption 2.2 hold. Then, $(\widehat{f}_k, \widehat{g}_k)$ in (2.18) is a (L_k, δ_k) inexact oracle of $f = \sum_{i=1}^N f_i$ at point $\overline{y}(k)$ with constants $L_k = 2NL$ and $\delta_k = L\|\widetilde{y}(k)\|^2$.

Lemma 2.6 implies that, if $L'_{k-1} = \frac{Nk}{c} \geq 2NL$, i.e., if $c \leq \frac{k}{2L}$, then the progress per iteration in Lemma 2.5 holds for (2.19) with $\delta_{k-1} := L\|\widetilde{y}(k-1)\|^2$. If $c \leq 1/(2L)$, Lemma 2.5 applies for *all iterations* $k = 1, 2, ...$; otherwise, it holds for all $k \geq 2cL$.

Proof: [Proof of Lemma 2.6] For notation simplicity, we re-write $y(k)$ and $\overline{y}(k)$ as y and $\overline{y}$, and $\widehat{f}_k, \widehat{g}_k, L_k, \delta_k$ as $\widehat{f}_y, \widehat{g}_y, L_y, \delta_y$. In view of Definition 4.11, we need to show inequalities (4.49). We first show the left one. By convexity of $f_i(\cdot)$: $f_i(x) \geq f_i(y_i) + \nabla f_i(y_i)^\top (x - y_i), \quad \forall x$; summing over $i = 1, ..., N$, using $f(x) = \sum_{i=1}^N f_i(x)$, and expressing $x - y_i = x - \overline{y} + \overline{y} - y_i$:

$$f(x) \geq \sum_{i=1}^N \Big(f_i(y_i) + \nabla f_i(y_i)^\top (\overline{y} - y_i) \Big) + \left(\sum_{i=1}^N \nabla f_i(y_i) \right)^\top (x - \overline{y}) = \widehat{f}_y + \widehat{g}_y^\top (x - \overline{y}).$$

We now prove the right inequality in (4.49). As $f_i(\cdot)$ is convex and has Lipschitz continuous derivative with constant L, we have: $f_i(x) \leq f_i(y_i) + \nabla f_i(y_i)^\top (x - y_i) + \frac{L}{2}\|x - y_i\|^2$, which, after summation over $i = 1, ..., N$, expressing $x - y_i = (x - \overline{y}) + (\overline{y} - y_i)$, and using the inequality $\|x - y_i\|^2 = \|(x - \overline{y}) + (\overline{y} - y_i)\|^2 \leq 2\|x - \overline{y}\|^2 + 2\|\overline{y} - y_i\|^2$, gives:

$$\begin{aligned} f(x) &\leq \sum_{i=1}^{N} \Big(f_i(y_i) + \nabla f_i(y_i)^\top (\overline{y} - y_i) \Big) + \left(\sum_{i=1}^{N} \nabla f_i(y_i) \right)^\top (x - \overline{y}) \\ &+ NL\|x - \overline{y}\|^2 + L \sum_{i=1}^{N} \|\overline{y} - y_i\|^2 = \widehat{f}_y + \widehat{g}_y^\top (x - \overline{y}) + \frac{2NL}{2}\|x - \overline{y}\|^2 + \delta_y, \end{aligned}$$

and so $(\widehat{f}_y, \widehat{g}_y)$ satisfy the right inequality in (4.49) with $L_y = 2NL$ and $\delta_y = L \sum_{i=1}^{N} \|\overline{y} - y_i\|^2$. □

Algorithm D–NC

Consider algorithm D–NC in (2.13)–(2.14). To avoid notational clutter, use the same notation as with D–NG for the global averages: $\overline{x}(k) := \frac{1}{N}\sum_{i=1}^{N} x_i(k)$, and $\overline{y}(k) := \frac{1}{N}\sum_{i=1}^{N} y_i(k)$, re-define $\widehat{f}_k, \widehat{g}_k$ for D–NC as in (2.18), and let $L'_{k-1} := \frac{N}{\alpha}$. Multiplying (2.13)–(2.14) from the left by $(1/N)\mathbf{1}^\top \otimes I$, and using $(\mathbf{1}^\top \otimes I)(W \otimes I) = \mathbf{1}^\top \otimes I$, we get that $\overline{x}(k), \overline{y}(k)$ satisfy (2.19). As $\alpha \leq 1/(2L)$, we have $L'_{k-1} \geq 2NL$, and so, by Lemma 2.6, the progress per iteration in Lemma 2.5 applies to $\overline{x}(k), \overline{y}(k)$ of D–NC for all k, with $\delta_{k-1} = L\|\widetilde{y}(k-1)\|^2$.

In summary, the analysis of the convergence rates of both D–NG and D–NC boils down to finding the disagreements $\|\widetilde{y}(k)\|$ and then applying Lemma 2.5.

2.5 Algorithm D–NG: Convergence analysis

This Section studies the convergence of D–NG. Subsection 2.5.1 bounds the disagreements $\|\widetilde{x}(k)\|$ and $\|\widetilde{y}(k)\|$ with D–NG; Subsection 2.5.2 combines these bounds with Lemma 2.5 to derive the convergence rate of D–NG and its dependence on the underlying network.

2.5.1 Algorithm D–NG: Disagreement estimate

This subsection shows that $\|\widetilde{x}(k)\|$ and $\|\widetilde{y}(k)\|$ are $O(1/k)$, hence establishing asymptotic consensus – the differences of the nodes' estimates $x_i(k)$ (and $y_i(k)$) converge to zero. Recall the step-size constant $c > 0$ in (2.8) and the gradient bound G in Assumption 2.3.

Theorem 2.7 (Consensus with D–NG) For D–NG in (2.6)–(2.8) under Assumptions 2.1 and 2.3:

$$\|\widetilde{x}(k)\| \leq \sqrt{N}\, c\, G\, C_{\text{cons}}\, \frac{1}{k} \quad \text{and} \quad \|\widetilde{y}(k)\| \leq 4\sqrt{N}\, c\, G\, C_{\text{cons}}\, \frac{1}{k}, \quad k = 1, 2, ..., \tag{2.20}$$

$$C_{\text{cons}} = \frac{8}{\sqrt{\kappa(1-\mu(W))}} \left\{ 2\,\mathcal{B}\left(\sqrt{\mu(W)}\right) + \frac{7}{1-\mu(W)} \right\}, \tag{2.21}$$

with $\mathcal{B}(r) := \sup_{z \geq 1/2} \left(z r^z \log(1+z)\right) \in (0, \infty)$, $r \in (0,1)$.

For notational simplicity, we prove Theorem 2.7 for $d = 1$, but the proof extends to a generic $d > 1$. We model the dynamics of the augmented state $(\widetilde{x}(k)^\top, \widetilde{x}(k-1)^\top)^\top$ as a linear time varying system with inputs $(I - J)\nabla F(y(k))$. We present here the linear system and solve it in the Appendix. Substitute the expression for $y(k-1)$ in (2.9); multiply the resulting equation from the left by $(I - J)$; use $(I - J)W = \widetilde{W} = \widetilde{W}(I - J)$; and set $\widetilde{x}(0) = 0$ by assumption. We obtain:

$$\begin{bmatrix} \widetilde{x}(k) \\ \widetilde{x}(k-1) \end{bmatrix} = \begin{bmatrix} (1+\beta_{k-2})\widetilde{W} & -\beta_{k-2}\widetilde{W} \\ I & 0 \end{bmatrix} \begin{bmatrix} \widetilde{x}(k-1) \\ \widetilde{x}(k-2) \end{bmatrix} - \alpha_{k-1} \begin{bmatrix} (I-J)\nabla F(y(k-1)) \\ 0 \end{bmatrix}, \tag{2.22}$$

for all $k = 1, 2, ...$, where β_k, for $k = 0, 1, ...$, is in (2.8), $\beta_{-1} = 0$, and $(\widetilde{x}(0)^\top, \widetilde{x}(-1)^\top)^\top = 0$. We emphasize that system (2.22) is more complex than the corresponding systems in, e.g., [2, 12], because the systems in [2, 12] involve only a single state $\widetilde{x}(k)$ and are not time varying (when W is constant); the upper bound on $\|\widetilde{x}(k)\|$ from (2.22) is an important technical contribution of this chapter; see Theorem 2.7 and Appendix A.

2.5.2 Convergence rate and network scaling

Theorem 2.8 (a) states the $O(\log k/k)$ convergence rate result for D–NG when the step-size constant $c \leq 1/(2L)$; Theorem 2.8 (b) (proved in Appendix A) demonstrates that the $O(\log k/k)$ convergence rate still holds if $c > 1/(2L)$, with a deterioration in the convergence constant. Part (b) assumes $x_i(0) = y_i(0) = 0$, $\forall i$, to avoid notational clutter.

Theorem 2.8 Consider the algorithm D–NG in (2.6)–(2.8) under Assumptions 2.1–2.3. Let $\|\overline{x}(0) - x^\star\| \leq R$, $R \geq 0$. Then:

(a) If $c \leq 1/(2L)$, we have, $\forall i$, $\forall k = 1, 2, ...$:

$$\frac{f(x_i(k)) - f^\star}{N} \leq \frac{2\,R^2}{c}\left(\frac{1}{k}\right) + 16\,c^2\;L\,C_{\rm cons}^2\,G^2\,\left(\frac{1}{k}\sum_{t=1}^{k-1}\frac{(t+2)^2}{(t+1)t^2}\right) + \;c\,G^2 C_{\rm cons}\,\left(\frac{1}{k}\right)$$
$$\leq \mathcal{C}\,\left(\frac{1}{k}\sum_{t=1}^{k}\frac{(t+2)^2}{(t+1)t^2}\right),\; \mathcal{C} = \frac{2R^2}{c} + 16c^2 L C_{\rm cons}^2 G^2 + \;c\,G^2 C_{\rm cons}. \quad (2.23)$$

(b) Let $x_i(0) = y_i(0) = 0, \forall i$. If $c > 1/(2L)$, (2.23) holds $\forall i$, $\forall k \geq 2\,cL$, with $\mathcal{C}$ replaced with $\mathcal{C}' = \mathcal{C}''(L, G, R, c) + 16c^2 L C_{\rm cons}^2 G^2 + \;c\,G^2 C_{\rm cons}$, and $\mathcal{C}''(L, G, R, c) \in [0, \infty)$ is a constant that depends on L, G, R, c, and *is independent of* $N, \mu(W)$.

We prove here Theorem 2.8 (a); for part (b), see Appendix A.

Proof: [Proof of Theorem 2.8 (a)] The proof consists of two parts. In Step 1 of the proof, we estimate the optimality gap $\frac{1}{N}(f(\overline{x}(k)) - f^\star)$ at the point $\overline{x}(k) = \frac{1}{N}\sum_{i=1}^N x_i(k)$ using Lemma 2.5 and the inexact oracle machinery. In Step 2, we estimate the optimality gap $\frac{1}{N}(f(x_i(k)) - f^\star)$ at any node i using convexity of the f_i's and the bound on $\|\widetilde{x}(k)\|$ from Theorem 2.7. □

Step 1. Optimality gap $(f(\overline{x}(k)) - f^\star)$

Recall that, for $k = 1, 2, ...$, $(\widehat{f}_k, \widehat{g}_k)$ in (2.18) is a (L_k, δ_k) inexact oracle of f at point $\overline{y}(k)$ with $L_k = 2NL$ and $\delta_k = L\|\widetilde{y}(k)\|^2$. Note that $(\widehat{f}_k, \widehat{g}_k)$ is also a (L'_k, δ_k) inexact oracle of f at point $\overline{y}(k)$ with $L'_k = N\frac{1}{c}(k+1) = \frac{N}{\alpha_k}$, because $\frac{1}{c} \geq 2L$, and so $L'_k \geq L_k$. Now, we apply Lemma 2.5 to (2.19), with $x^\bullet = x^\star$, and the Lipschitz constant $L'_k = 1/(\alpha_k/N)$. Recall that $\overline{v}(k) = \frac{\overline{y}(k) - (1-\gamma_k)\overline{x}(k)}{\gamma_k}$. We get:

$$\begin{aligned}\frac{(k+1)^2}{k}\,(f(\overline{x}(k)) - f^\star) \;&+\; \frac{2N}{c}\|\overline{v}(k) - x^\star\|^2 \qquad (2.24)\\ &\leq\; \frac{k^2-1}{k}\,(f(\overline{x}(k-1)) - f^\star) + \frac{2N}{c}\|\overline{v}(k-1) - x^\star\|^2 + L\|\widetilde{y}(k-1)\|^2\frac{(k+1)^2}{k}.\end{aligned}$$

Because $\frac{(k+1)^2}{k} \geq \frac{(k+1)^2-1}{k+1}$, and $(f(\overline{x}(k)) - f^\star) \geq 0$, we have:

$$\begin{aligned}&\frac{(k+1)^2-1}{k+1}\,(f(\overline{x}(k)) - f^\star) + \frac{2N}{c}\|\overline{v}(k) - x^\star\|^2\\ \leq\;& \frac{k^2-1}{k}\,(f(\overline{x}(k-1)) - f^\star) + \frac{2N}{c}\|\overline{v}(k-1) - x^\star\|^2 + L\|\widetilde{y}(k-1)\|^2\frac{(k+1)^2}{k}.\end{aligned}$$

By unwinding the above recursion, and using $\overline{v}(0) = \overline{x}(0)$, gives: $\frac{(k+1)^2-1}{k+1}\,(f(\overline{x}(k)) - f^\star) \leq \frac{2N}{c}\|\overline{x}(0) - x^\star\|^2 + L\sum_{t=1}^k \|\widetilde{y}(t-1)\|^2\frac{(t+1)^2}{t}$. Applying Theorem 2.7 to the last equation, and using $\frac{k+1}{(k+1)^2-1} =$

$\frac{k+1}{k(k+2)} \leq \frac{k+2}{k(k+2)} = \frac{1}{k}$, and the assumption $\|\widetilde{y}(0)\| = 0$, leads to, as desired:

$$(f(\overline{x}(k)) - f^\star) \leq \frac{1}{k}\frac{2N}{c}\|\overline{x}(0) - x^\star\|^2 + \frac{16\,c^2\,N}{k} L\, C_{\rm cons}^2 G^2 \sum_{t=2}^{k} \frac{(t+1)^2}{t(t-1)^2}. \tag{2.25}$$

Optimality gap $(f(x_i(k)) - f^\star)$

Fix an arbitrary node i; then, by convexity of f_j, $j = 1, 2, ..., N$: $f_j(\overline{x}(k)) \geq f_j(x_i(k)) + \nabla f_j(x_i(k))^\top (\overline{x}(k) - x_i(k))$, and so: $f_j(x_i(k)) \leq f_j(\overline{x}(k)) + G\|\overline{x}(k) - x_i(k)\|$. Summing the above inequalities for $j = 1, ..., N$, using $\sum_{i=1}^N \|\overline{x}(k) - x_i(k)\| = \sum_{i=1}^N \|\widetilde{x}_i(k)\| \leq \sqrt{N}\|\widetilde{x}(k)\|$, subtracting $f^\star$ from both sides, and using $\|\widetilde{x}(k)\| \leq \sqrt{N} c G C_{\rm cons}(1/k)$ from Theorem 2.7:

$$f(x_i(k)) - f^\star \;\; \leq \;\; f(\overline{x}(k)) - f^\star + G\sqrt{N}\|\widetilde{x}(k)\| \leq f(\overline{x}(k)) - f^\star + c\,N\,C_{\rm cons} G^2 \frac{1}{k}, \tag{2.26}$$

which, with (2.25) where the summation variable t is replaced by $t+1$, completes the proof. □

Network Scaling

Using Theorem 2.8, Theorem 2.9 studies the dependence of the convergence rate on the underlying network – N and W, when: 1) nodes do not know L and $\mu(W)$ before the algorithm run, and they set the step-size constant c to a constant independent of N, L, W, e.g., $c = 1$; and 2) nodes know $L, \mu(W)$, and they set $c = \frac{1-\mu(W)}{2L}$. See [12] for dependence of $1/(1-\mu(W))$ on N for commonly used models, e.g., expanders or geometric graphs.

Theorem 2.9 Consider the algorithm D–NG in (2.6)–(2.8) under Assumptions 2.1–2.3. Then:

(a) For arbitrary $c = \text{const} > 0$: $\frac{1}{N}\left(f(x_i(k)) - f^\star\right) = O\left(\frac{1}{(1-\mu(W))^{3+\xi}} \frac{\log k}{k}\right)$.

(b) For $c = \frac{1-\mu(W)}{2L}$: $\frac{1}{N}\left(f(x_i(k)) - f^\star\right) = O\left(\frac{1}{(1-\mu(W))^{1+\xi}} \frac{\log k}{k}\right)$.

Proof: [Proof of Theorem 2.9] Fix $\kappa \in (0,1)$ and $\xi \in (0,1)$ (two arbitrarily small positive constants). By Assumption 2.1 (b), $\mu = \mu(W) \in [\kappa, 1]$. We show that for $C_{\rm cons}$ in (2.21):

$$C_{\rm cons} \leq A(\xi, \kappa)\; \frac{1}{(1-\mu)^{3/2+\xi}},\;\; \forall \mu \in [\kappa, 1], \tag{2.27}$$

where $A(\xi,\kappa) \in (0,\infty)$ depends only on ξ, κ. Consider $\mathcal{B}(r) = \sup_{z\geq 1/2}\{z\, r^z \log(1+z)\}$, $r \in (0,1)$; there exists $K_B(\xi) \in (0,\infty)$ such that: $\log(1+z) \leq K_B(\xi) z^\xi$, $\forall z \geq 1/2$. Thus:

$$\mathcal{B}(r) \leq K_B(\xi) \sup_{z\geq 1/2}\left\{z^{1+\xi} r^z\right\} = K_B(\xi)\, e^{-(1+\xi)}(1+\xi)^{(1+\xi)} \frac{1}{(-\log r)^{1+\xi}} =: \frac{A'(\xi)}{(-\log r)^{1+\xi}},$$

for all $r \in (0,1)$. From the above equation, and using $1/(-\log\sqrt{u}) \leq 2/(1-u)$, $\forall u \in [0,1)$, we have $\mathcal{B}\left(\sqrt{\mu}\right) \leq 2A'(\xi)/(1-\mu)^{1+\xi}$. The latter, applied to (2.21), yields (2.27) , with

$$A(\xi,\kappa) := \frac{8}{\sqrt{\kappa}} \max\left\{3A'(\xi), 7\right\}.$$

Claim (a) now follows by taking arbitrary $c = \Theta(1)$ and applying (2.27) to Theorem 2.8 (b); and claim (b) follows by taking $c = \frac{1-\mu}{2L}$ and applying (2.27) to Theorem 2.8 (a). □

2.6 Algorithm D–NC: Convergence analysis

We now consider the D–NC algorithm. Subsection 2.6.1 provides the disagreement estimate, while Subsection 2.6.1 gives the convergence rate and network scaling.

2.6.1 Disagreement estimate

We estimate the disagreements $\widetilde{x}(k)$, and $\widetilde{y}(k)$ with D–NC.

Theorem 2.10 (Consensus with D–NC) Let Assumptions 2.1 (a) and 2.3 hold, and consider the algorithm D–NC. Then, for $k = 1, 2, ...$:

$$\|\widetilde{x}(k)\| \leq 2\alpha\sqrt{N}G\frac{1}{k^2} \quad \text{and} \quad \|\widetilde{y}(k)\| \leq 2\alpha\sqrt{N}G\frac{1}{k^2}. \tag{2.28}$$

For notational simplicity, we perform the proof for $d = 1$, but it extends to a generic $d > 1$. Denote by $B_{t-1} := \max\{\|\widetilde{x}(t-1)\|,\ \|\widetilde{y}(t-1)\|\}$, and fix $t-1$. We want to upper bound B_t. Multiplying (2.13)–(2.14) by $(I-J)$ from the left, using $(I-J)W = \widetilde{W}(I-J)$:

$$\widetilde{x}(t) = \widetilde{W}^{\tau_x(t)}\,\widetilde{y}(t-1) - \alpha\widetilde{W}^{\tau_x(t)}(I-J)\nabla F(y(t-1)) \tag{2.29}$$

$$\widetilde{y}(t) = \widetilde{W}^{\tau_y(t)}\left[\ \widetilde{x}(t) + \beta_{t-1}(\widetilde{x}(t) - \widetilde{x}(t-1))\ \right]. \tag{2.30}$$

We upper bound $\|\widetilde{x}(t)\|$ and $\|\widetilde{y}(t)\|$ from (2.29), (2.30). Recall $\|\widetilde{W}\| = \mu(W) := \mu \in (0,1)$; from (2.11)

and (2.12), we have $\mu^{\tau_x(t)} \leq \frac{1}{t^2}$ and $\mu^{\tau_y(t)} \leq \frac{1}{3t^2}$. From (2.29), using the sub-additive and sub-multiplicative properties of norms, and using $\|\widetilde{y}(t-1)\| \leq B_{t-1}$, $\mu \in (0,1)$, $\|(I-J)\nabla F(y(t-1))\| \leq \|\nabla F(y(t-1))\| \leq \sqrt{N}G$, $\beta_{t-1} \leq 1$:

$$\begin{aligned}
\|\widetilde{x}(t)\| &\leq \mu^{\tau_x(t)} B_{t-1} + \alpha\mu^{\tau_x(t)}\sqrt{N}G \leq \frac{1}{t^2} B_{t-1} + \alpha\sqrt{N}G\frac{1}{t^2} \qquad (2.31)\\
\|\widetilde{y}(t)\| &\leq 2\,\mu^{\tau_y(t)}\|\widetilde{x}(t)\| + \mu^{\tau_y(t)}\|\widetilde{x}(t-1)\| \\
&\leq 2\,\mu^{\tau_x(t)+\tau_y(t)} B_{t-1} + 2\alpha\sqrt{N}G\,\mu^{\tau_x(t)+\tau_y(t)} + \mu^{\tau_y(t)} B_{t-1} \\
&\leq 3\,\mu^{\tau_y(t)} B_{t-1} + 2\alpha\sqrt{N}G\mu^{\tau_y(t)} \leq \frac{1}{t^2} B_{t-1} + \alpha\sqrt{N}G\frac{1}{t^2}. \qquad (2.32)
\end{aligned}$$

Clearly, from (2.31) and (2.32):$B_t \leq \frac{1}{t^2} B_{t-1} + \frac{1}{t^2}\alpha\sqrt{N}G$. Next, using $B_0 = 0$, unwind the latter recursion for $k = 1, 2$, to obtain, respectively: $B_1 \leq \alpha\sqrt{N}G$ and $B_2 \leq \alpha\sqrt{N}G/2$, and so (2.28) holds for $k = 1, 2$. Further, for $k \geq 3$ unwinding the same recursion for $t = k, k-1, ..., 1$:

$$\begin{aligned}
B_k &\leq \frac{\alpha\sqrt{N}G}{k^2}\left(1 + \sum_{t=2}^{k-1} \frac{1}{(k-1)^2(k-2)^2...t^2} + \frac{1}{(k-1)^2(k-2)^2...2^2}\right) \\
&\leq \frac{\alpha\sqrt{N}G}{k^2}\left(1 + \sum_{t=2}^{k-1}\frac{1}{t^2} + \frac{1}{2^2}\right) \leq \frac{\alpha\sqrt{N}G}{k^2}\left(\frac{\pi^2}{6} + \frac{1}{4}\right) \leq \frac{2\,\alpha\sqrt{N}G}{k^2},\ \ k = 1, 2, ...,
\end{aligned}$$

where we use $1 + \sum_{t=2}^{k-1} \frac{1}{t^2} \leq \pi^2/6$, $\forall k \geq 3$. □

2.6.2 Convergence rate and network scaling

We are now ready to state the Theorem on the convergence rate of D–NC.

Theorem 2.11 Consider the algorithm D–NC under Assumptions 2.1 (a), 2.2, and 2.3. Let $\|\overline{x}(0) - x^\star\| \leq R$, $R \geq 0$. Then, after $\mathcal{K} = \sum_{t=1}^{k} (\tau_x(t) + \tau_y(t)) \leq \frac{2}{-\log \mu(W)} (k \log 3 + 2(k+1)\log(k+1)) = O(k \log k)$ communication rounds, i.e., after k outer iterations, at any node i:

$$\frac{1}{N}\left(f(x_i(k)) - f^\star\right) \leq \frac{1}{k^2}\left(\frac{2}{\alpha}R^2 + 11\,\alpha^2 LG^2 + \alpha G^2\right), \quad k = 1, 2, ... \qquad (2.33)$$

[Proof outline] The proof is very similar to the proof of Theorem 2.8 (a) (for details see the second version v2 on arxiv [9]); first upper bound $f(\overline{x}(k)) - f^\star$, and then $f(x_i(k)) - f^\star$. To upper bound $f(\overline{x}(k)) - f^\star$, recall that the evolution (2.19) with $\alpha_k = \alpha$ for $(\overline{x}(k), \overline{y}(k))$ is the inexact Nesterov gradient with the inexact oracle $(\widehat{f}_k, \widehat{g}_k)$ in (2.18), and $(L_k = 2NL,\ \delta_k = L\|\widetilde{y}(k)\|^2)$. Then, apply Lemma 2.5 with $x^\bullet \equiv x^\star$ and $L'_{k-1} = N/\alpha$, and use the bound on $\|\widetilde{y}(k)\|$ from Theorem 2.10, to obtain the bound on $f(\overline{x}(k)) - f^\star$.

Finally, find the bound on $f(x_i(k)) - f^\star$ analogously to the proof of Theorem 2.8 (a).

Network scaling

We now give the network scaling for algorithm D–NC in Theorem 2.12. We assume that nodes know L and $\mu(W)$ before the algorithm run.

Theorem 2.12 Consider D–NC under Assumptions 2.1 (a), 2.2, and 2.3 with step-size $\alpha \leq 1/(2L)$. Then, after $\mathcal{K}$ communication rounds, at any node i, $\frac{1}{N}\left(f(x_i) - f^\star\right)$ is $O\left(\frac{1}{(1-\mu(W))^2}\frac{1}{\mathcal{K}^{2-\xi}}\right)$.

Proof: Fix $\xi \in (0,1)$, and let $\mathcal{K}$ be the number of elapsed communication rounds after k outer iterations. There exists $C_0(\xi) \in (1,\infty)$, such that, $2\left(k\log 3 + 2(k+1)\log(k+1)\right) \leq C_0(\xi)k^{1+\xi}$, $\forall k \geq 1$. The latter, combined with $1/(-\log\mu(W)) \leq 1/(1-\mu(W))$, $\mu(W) \in [0,1)$, and the upper bound bound on $\mathcal{K}$ in Theorem 2.11, gives: $1/k^2 \leq \left(C_0(\xi)\right)^2 \frac{1}{(1-\mu(W))^2}\frac{1}{\mathcal{K}^{2-2\xi}}$. Plugging the latter in the optimality gap bound in Theorem 2.11, the result follows (replacing ξ with $\xi/2$.) $\square$

2.7 Comparisons with the literature and discussion of the Assumptions

Subsection 2.7.1 compares D–NG, D–NC, and the distributed (sub)gradient algorithms in [2, 12, 29], from the aspects of implementation and convergence rate; Subsection 2.7.2 gives a detailed discussion on Assumptions 2.1–2.3.

2.7.1 Comparisons of D–NG and D–NC with the literature

We first set up the comparisons by explaining how to account for Assumption 2.1 (b) and by adapting the results in [29, 30] to our framework.

Assumption 2.1 (b). We account for Assumption 2.1 (b) with D–NG as follows. Suppose that the nodes are given arbitrary symmetric, doubly stochastic weights W with $\mu(W) < 1$ – the matrix required by D–NC and [2, 12, 29]. (For example, the Metropolis weights W.) As the nodes may not be allowed to check whether the given W obeys Assumption 2.1 (b) or not, they modify the weights to $W' := \frac{1+\kappa}{2}I + \frac{1-\kappa}{2}W$, where $\kappa \in (0,1)$ can be taken arbitrarily small. The matrix W' obeys Assumption 2.1 (b), whether W obeys it or not.[5] The modification is done without any required knowledge of the system parameters nor inter-node communication; node i sets: 1) $W'_{ij} = \frac{1-\kappa}{2}W_{ij}$, for $\{i,j\} \in E$, $i \neq j$; 2) $W'_{ij} = 0$, for $\{i,j\} \notin E$, $i \neq j$; and 3) $W'_{ii} := 1 - \sum_{j\neq i} W'_{ij}$. To be fair, when we compare D–NG with other methods

[5]Note that $\lambda_1(W') \geq \frac{1+\kappa}{2} - \frac{1-\kappa}{2}|\lambda_1(W)| \geq \kappa$, because $|\lambda_1(W)| < 1$, and so $W' \succeq \kappa\, I$ whether $W \succeq \kappa\, I$ or not.

(either theoretically as we do here or numerically as done in Section 2.9), we set its weights to W'. For theoretical comparisons, from Theorem 2.8, the convergence rate of D–NG depends on W' through the inverse spectral gap $1/(1-\mu(W'))$. It can be shown that $\frac{1}{1-\mu(W')} = \frac{2}{1-\kappa}\frac{1}{1-\mu(W)}$, i.e., the spectral gaps of W and W' differ only by a constant factor and the weight modification does not affect the convergence rate (up to a numerical constant); henceforth, we express the theoretical rate for D–NG in terms of W.

References [29, 30] develop and analyze non-accelerated and accelerated distributed gradient and proximal gradient methods for time-varying networks and convex f_i's that have a differentiable component with Lipschitz continuous and bounded gradient and a non-differentiable component with bounded gradient. To compare with reference [30], we adapt it to our framework of static networks and differentiable f_i's. (We set the non-differentiable components of the f_i's to zero.) The accelerated methods in [30] achieve faster rates than the non-accelerated ones; we focus only on the former. References [29, 30] assume deterministic time-varying networks. To adapt their results to our static network setup in a fair way, we replace the parameter γ in [29] (see equation (7) in [29]) with $\mu(W)$. The references propose two variants of the accelerated algorithm: the first (see (6a)–(6d) in [29]) has k inner consensus iterations at the outer iteration k, while the second one has $\lceil 4\log(k+1)/(-\log\mu)\rceil$ (See Subsection III-C in [29].) The bounds established in [29] for the second variant give its rate: 1) $O\left(\frac{N^2}{(1-\mu(W))^2\mathcal{K}^{2-\xi}}\right)$, when nodes know only $\mu(W)$ and L; and 2) $O\left(\frac{N^{1/2}}{(1-\mu(W))^2\mathcal{K}^{2-\xi}}\right)$, when they in addition know N and set the step-size $\alpha = \Theta(1/\sqrt{N})$. The first variant has a slower rate (see Appendix A).

Algorithm implementation and convergence rate

Table 2.1 compares D–NG, D–NC, the algorithm in [12] and the second algorithm in [29] with respect to implementation and the number of communications $\mathcal{K}(\epsilon; N, W)$ to achieve ϵ-accuracy. Here $\mathcal{K}(\epsilon; N, W)$ is the smallest number of communication rounds $\mathcal{K}$ after which $\frac{1}{N}(f(x_i) - f^\star) \leq \epsilon, \forall i$. Regarding implementation, we discuss the knowledge required a priori by all nodes for: 1) convergence (row 1); and 2) stopping and optimizing the step-size (row 2). By stopping, we mean determining a priori the (outer) iteration k_0 such that $\frac{1}{N}(f(x_i(k)) - f^\star) \leq \epsilon, \forall k \geq k_0, \forall i$. Optimizing the step size here means finding the step-size that minimizes the established upper bound (in the reference of interest) on the optimality gap (e.g., the bound for D–NG in Theorem 2.8 (a).) We assume, with all methods, that W is already given (e.g., Metropolis.) Regarding $\mathcal{K}(\epsilon; N, W)$, we neglect the logarithmic and ξ-small factors and distinguish two cases: 1) the nodes have no global knowledge (row 3); and 2) the nodes know $L, \mu(W) =: \mu$. We can see from Table 2.1 that, without global knowledge (row 3), D–NG has better dependence on ϵ than [12] and worse dependence on μ. Under global knowledge (row 4), D–NC has the best complexity. When with the second method

	D–NG	D–NC	[12]	[29]
Knowledge for convergence	none	L, μ	none	L, μ
Knowledge for stopping; stepsize	μ, R, G, L	μ, R, G, L	μ, R, G	μ, R, G, L, N
$\mathcal{K}(\epsilon; N, W)$: No knowledge	$O\left(\frac{1}{(1-\mu)^3\epsilon}\right)$	not guaranteed	$O\left(\frac{1}{(1-\mu)^2\epsilon^2}\right)$	not studied
$\mathcal{K}(\epsilon; N, W)$: L, μ	$O\left(\frac{1}{(1-\mu)\epsilon}\right)$	$O\left(\frac{1}{(1-\mu)\sqrt{\epsilon}}\right)$	$O\left(\frac{1}{(1-\mu)\epsilon^2}\right)$	$O\left(\frac{N}{(1-\mu)\sqrt{\epsilon}}\right)$

Table 2.1: Comparisons of algorithms D–NG, D–NC, [12], and [29] (algorithms 1 and 2).

in [29] nodes in addition know N, their bound improves to $O\left(\frac{N^{1/4}}{(1-\mu)\sqrt{\epsilon}}\right)$ (see Appednix A.) Further, while D–NG and [12] require no knowledge of any global parameters for convergence (row 1), D–NC and the second algorithm in [29] need L and $\mu(W)$. The first variant in [29] requires only L. Also, the bounds in [12] in Table 2.1 hold for a wider class of functions; see [12] for details.

Global knowledge $\mu(W), L, G, R$

Global knowledge $\mu(W), L, G, R$ (as needed, e.g., by D–NG for stopping) can be obtained as follows. Consider first L (see Assumption 2.2) and suppose that each node knows a Lipschitz constant L_i of its own f_i. Then, L can be taken as $L = \max_{i=1,\ldots,N} L_i$. Thus, each node can compute L if nodes run a distributed algorithm for maximum computation, e.g., ([74], (1)); all nodes get L after $O(\mathrm{Diam})$ per-node communicated scalars, where Diam is the network diameter. Likewise, a gradient bound G (see Assumption 2.3) can be taken as $G = \max_{i=1,\ldots,N} G_i$, where G_i is a gradient bound for the f_i. The quantity $\mu(W)$ (equal to the second largest eigenvalue of W) can be computed in a distributed way, e.g., by algorithm DECENTRALOI, proposed for a more general setting in [75], and adapted to the problem like ours in [[10], Subsection IV-A, p. 2519]. With DECENTRALOI, node i obtains q_i^μ, the i-th coordinate of the $N \times 1$ eigenvector q^μ of W that corresponds to $\mu(W)$, (up to ϵ-accuracy) after $O\left(\frac{\log^2(N/\epsilon)\log N}{1-\mu(W)}\right)$ per-node communicated scalars [75]; then, node i obtains $\mu(W)$ as: $\frac{\sum_{j\in O_i} W_{ij} q_j^\mu}{q_i^\mu}$.

Consider now D–NC when nodes do not have available their local gradient Lipschitz constants L_i. Nodes can take a diminishing step size $\alpha_k = 1/(k+1)^p$, $p \in (0, 1]$, and still guarantee convergence, with a deteriorated rate $O\left(\frac{1}{\mathcal{K}^{2-p-\xi}}\right)$. In alternative, it may be possible to employ a "distributed line search," similarly to [76]. Namely, in the absence of knowledge of the gradient's Lipschitz constant L, the centralized Nesterov gradient method with a backtracking line search achieves the same rate $O(1/k^2)$, with an additional computational cost per iteration k; see [73, 77]. It is an interesting research direction to develop a variant of distributed line search for D–NC type methods and explore the amount of incurred additional communications/computations per outer iteration k; due to lack of space, this is left for future work.

The $\Omega(1/k^{2/3})$ lower bound on the worst-case optimality gap for [2]

We now focus on the dependence of the convergence rate on k and $\mathcal{K}$ only (assuming a finite, fixed $1/(1-\mu(W))$.) We demonstrate that D–NG has a strictly better worst-case convergence rate in k (and $\mathcal{K}$) than [2], when applied to the f_i's defined by Assumptions 2.2 and 2.3. Thus, D–NC also has a better rate.

Fix a generic, connected network $\mathcal{G}$ with N nodes and a weight matrix W that obeys Assumption 2.1. Let $\mathcal{F}=\mathcal{F}(L,G)$ be the class of all N-element sets of functions $\{f_i\}_{i=1}^N$, such that: 1) each $f_i:\mathbb{R}^d\to\mathbb{R}$ is convex, has Lipschitz continuous derivative with constant L, and bounded gradient with bound G; and 2) problem (2.1) is solvable in the sense of Assumption 2.2 (a). Consider (2.1) with $\{f_i\}_{i=1}^N\in\mathcal{F}$, for all i; consider D–NG with the step-size $\alpha_k=\frac{c}{(k+1)}$, $k=0,1,...$, $c\leq 1/(2L)$. Denote by: $\mathcal{E}^{\mathrm{D-NG}}(k,R)=\sup_{\{f_i\}_{i=1}^N\in\mathcal{F}}\sup_{\{\overline{x}(0):\,\|\overline{x}(0)-x^\star\|\leq R\}}\max_{i=1,...,N}\{f(x_i(k))-f^\star\}$ the optimality gap at the k-th iteration of D–NG for the worst $\{f_i\}_{i=1}^N\in\mathcal{F}$, and the worst $\overline{x}(0)$ (provided $\|\overline{x}(0)-x^\star\|\leq R$.) From Theorem 2.8, *for any* $k=1,2,...$: $\mathcal{E}^{\mathrm{D-NG}}(k,R)\leq\mathcal{C}\frac{\log k}{k}=O(\log k/k)$, with $\mathcal{C}$ in (2.23). Now, consider the algorithm in [2] with the step-size $\alpha_k=\frac{c}{(k+1)^\tau}$, $k=0,1,...$, where $c\in[c_0,1/(2L)]$, $\tau\geq 0$ are the degrees of freedom, and c_0 is an arbitrarily small positive number. With this algorithm, $k=\mathcal{K}$. We show that, for the $N=2$-node connected network, the weight matrix W with $W_{ii}=7/8$, $i=1,2$, and $W_{ij}=1/8$, $i\neq j$ (which satisfies Assumption 2.1), and $R=\sqrt{2}$, $L=\sqrt{2}$ and $G=10$, with [2]:

$$\inf_{\tau\geq 0,\,c\in[c_0,1/(2L)]}\mathcal{E}(k,R;\tau,c)=\Omega\left(\frac{1}{k^{2/3}}\right),\tag{2.34}$$

where $\mathcal{E}(k,R;\tau,c)=\sup_{\{f_i\}_{i=1}^N\in\mathcal{F}}\sup_{\{\overline{x}(0):\,\|\overline{x}(0)-x^\star\|\leq R\}}\max_{i=1,...,N}\{f(x_i(k))-f^\star\}$ is the worst-case optimality gap when the step-size $\alpha_k=\frac{c}{(k+1)^\tau}$ is used. We perform the proof by constructing a "hard" example of the functions $f_i\in\mathcal{F}(L,G)$ and a "hard" initial condition to upper bound $\mathcal{E}(k,R;\tau,c)$; for any fixed k,c,τ, we set: $x_i(0)=:(1,0)^\top$, $i=1,2$; $f_i=:f_i^{\theta_k}$, where:

$$f_i^\theta(x)=\begin{cases}\frac{\theta(x^{(1)}+(-1)^i)^2}{2}+\frac{(x^{(2)}+(-1)^i)^2}{2} & \text{if } \theta(x^{(1)}+(-1)^i)^2+(x^{(2)}+(-1)^i)^2\leq\overline{\chi}^2\\ \overline{\chi}\left(\left[\theta(x^{(1)}+(-1)^i)^2+(x^{(2)}+(-1)^i)^2\right]^{1/2}-\frac{\overline{\chi}}{2}\right) & \text{else;}\end{cases}\tag{2.35}$$

$\theta_k=\frac{1}{\sum_{t=0}^{k-1}(t+1)^{-\tau}}$; and $\overline{\chi}=6$. The proof of (2.34) is in the Appendix. We convey here the underlying intuition. When τ is ϵ-smaller (away) from one, we show:

$$\max_{i=1,2}(f^{\theta_k}(x_i(k))-f^{\star,\theta_k})\geq\Omega\left(\frac{1}{k^{1-\tau}}+\frac{1}{k^{2\tau}}\right).$$

The first summand is the "optimization term," for which a counterpart exists in the centralized gradient

method also. The second, "distributed problem" term, arises because the gradients $\nabla f_i(x^\star)$ of the individual nodes functions are non-zero at the solution $x^\star$. Note the two opposing effects with respect to τ: $\frac{1}{k^{1-\tau}}$ (the smaller $\tau \geq 0$, the better) and $\frac{1}{k^{2\tau}}$ (the larger $\tau \geq 0$, the better.) To balance the opposing effects of the two summands, one needs to take a diminishing step-size; $\tau = 1/3$ strikes the needed balance to give the $\Omega(1/k^{2/3})$ bound.

2.7.2 Discussion on Assumptions

We now discuss what may occur if we drop each of the Assumptions made in our main results–Theorems 2.7 and 2.8 for D–NG, and Theorems 2.10 and 2.11 for D–NC.

Assumption 2.1 (a)

Consider Theorems 2.7 and 2.10. If Assumption 2.1 (a) is relaxed, then $\widetilde{x}(k)$ with both methods may not converge to zero. Similarly, consider Theorems 2.8 and 2.11. Without Assumption 2.1 (a), $f(x_i(k))$ may not converge to $f^\star$ at any node. Consider the following simple example: $N = 2$, $W = I$; let $f_i : \mathbb{R} \to \mathbb{R}$, $i = 1, 2$, obey Assumptions 2.2 and 2.3, with the three quantities: $x_i^\star := \arg\min_{x\in\mathbb{R}} f_i(x)$, $i = 1, 2$, and $x^\star = \arg\min_{x\in\mathbb{R}} [\, f(x) = f_1(x) + f_2(x) \,]$ all unique and mutually different; set arbitrary intialization $x(0) = y(0)$. Then, $x_i(k)$ converges to $x_i^\star$ (by convergence of the centralized Nesterov gradient method,) while $\widetilde{x}(k)$ and $f(x_i(k)) - f^\star$, $i = 1, 2$, converge respectively to the non-zero values: $(\frac{x_1^\star - x_2^\star}{2}, \frac{x_2^\star - x_1^\star}{2})^\top$ and $f(x_i^\star) - f^\star$, $i = 1, 2$.

Assumption 2.1 (b)

Assumption 2.1 (b) is imposed only for D–NG – Theorems 2.7 and 2.8. We show by simulation that, if relaxed, $\|\widetilde{x}(k)\|$ and $f(x_i(k)) - f^\star$ may grow unbounded. Take $N = 2$ and $W_{11} = W_{22} = 1/10$, $W_{12} = W_{21} = 9/10$; the Huber losses $f_i : \mathbb{R} \to \mathbb{R}$, $f_i(x) = \frac{1}{2}(x - a_i)^2$ if $\|x - a_i\| \leq 1$ and $f_i(x) = \|x - a_i\| - 1/2$ else, $a_i = (-1)^{i+1}$; $c = 1$, and $x(0) = y(0) = (0, 0)^\top$. Then, we verify by simulation (see Figure 2.1) that $\|\widetilde{x}(k)\|$ and $\min_{i=1,2}(\, f(x_i(k)) - f^\star \,)$ grow unbounded.

Assumption 2.2

Assumption 2.2 is not needed for consensus with D–NG and D–NC (Theorems 2.7 and 2.10), but we impose it for Theorems 2.8 and 2.11 (convergence rates of D–NG and D–NC). This Assumption is standard and widely present in the convergence analysis of gradient methods, e.g., [11]. Nonetheless, we consider what

may occur if we relax the requirement on the Lipschitz continuity of the gradient of the f_i's. For both D–NG and D–NC, we borrow the example functions $f_i : \mathbb{R} \to \mathbb{R}$, $i = 1, 2$, from [30], pages 29–31: $f_1(x) = 4x^3 + \frac{3x^2}{2}$, $x \geq 1$; $f_1(x) = \frac{15x^2}{2} - 2$, $x < 1$; and $f_2(x) := f_1(-x)$. Then, for D–NG with $W_{11} = W_{22} = 1 - W_{12} = 1 - W_{21} = 9/10$, $c = 1$, and $x(0) = y(0) = (-1, 1)^\top$, simulations show that $\|x(k)\|$ and $f(x_i(k)) - f^\star$, $i = 1, 2$, grow unbounded. Similarly, with D–NC, for the same W, $\alpha = 0.1$, and $x(0) = y(0) = (-1, 1)^\top$, simulations show that $f(x_i(k)) - f^\star$, $i = 1, 2$, stays away from zero when k grows (see Figure 2.1.)

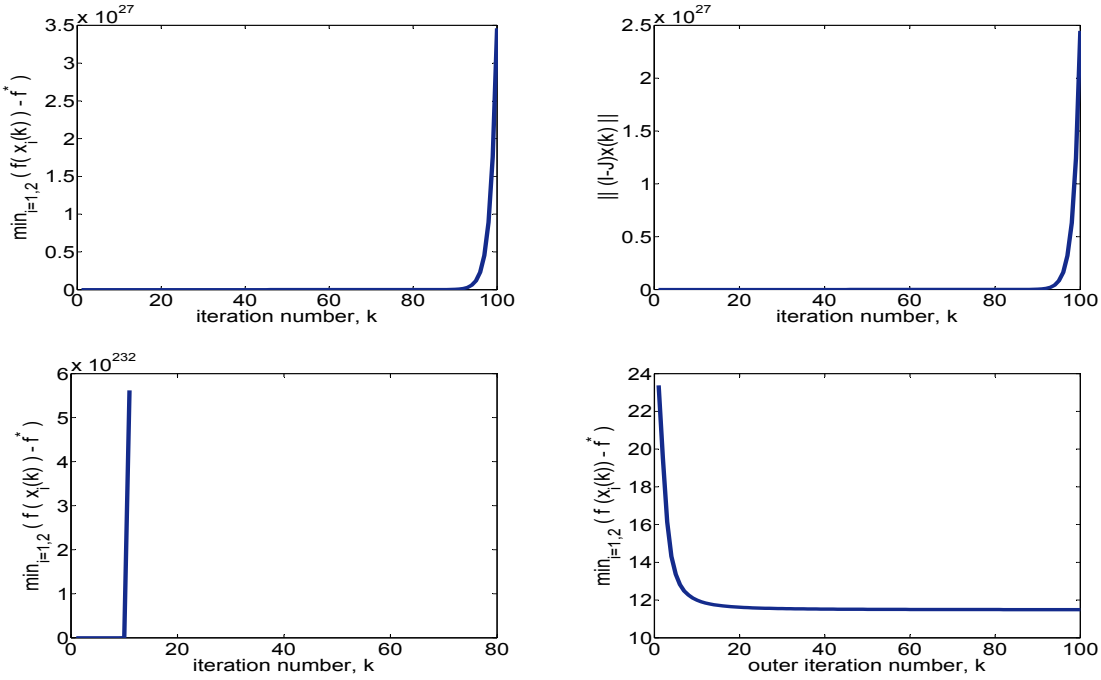

Figure 2.1: Top: Divergence of D–NG for the $N = 2$-node network with W that violates Assumption 1 (b); Left: $\min_{i=1,2}(f(x_i(k)) - f^\star)$; Right: $\|\widetilde{x}(k)\| = \|(I - J)x(k)\|$; Bottom: Example f_i's, $i = 1, 2$, in reference [30], pages 29–31 that do not have the Lipschitz continuous gradients; Left: D–NG diverges; Right: D–NC does not converge to a solution – $\min_{i=1,2}(f(x_i(k)) - f^\star)$ does not converge to zero.

Assumption 2.3

First consider Theorems 2.8 and 2.11 on the convergence rates of D–NG and D–NC. Define the class $\overline{\mathcal{F}}(L)$ to be the collection of all N-element sets of convex functions $\{f_i\}_{i=1}^N$, where each $f_i : \mathbb{R}^d \to \mathbb{R}$ has Lipschitz continuous gradient with constant L, and problem (2.2) is solvable in the sense of Assumption 2.2 (a). (Assumption 2.3 relaxed.) With the D–NC for the 2-node connected network, arbitrary weight matrix W obeying Assumption 2.1 (a), and the step-size $\alpha = 1/(2L)$, we show for $L = 1$, $R \geq 0$, that, for any $k \geq 5$

and arbitrarily large $M > 0$:

$$\mathcal{E}(k; R; \alpha = 1/(2L)) = \sup_{\{f_i\}_{i=1}^N \in \overline{\mathcal{F}}(L=1)} \sup_{\overline{x}(0): \|\overline{x}(0) - x^\star\| \leq R} \max_{i=1,2} (f(x_i(k)) - f^\star) \geq M. \tag{2.36}$$

Note that the above means $\mathcal{E}(k; R; \alpha = 1/(2L)) = +\infty$, $\forall k \geq 10$, $\forall R \geq 0$. That is, no matter how large the (outer) iteration number k is, the worst case optimality gap is still arbitrarily large.

We conduct the proof by making a "hard" instance for $\{f_i\}_{i=1}^N$: for a fixed k, M, we set $x_i(0) = y_i(0) = 0$, $i = 1, 2$, $f_i : \mathbb{R} \to \mathbb{R}$, to $f_i = f_i^{\theta(k,M)}$, where $\theta = \theta(k, M) = 8\sqrt{M}\, k^2$ and:

$$f_i^\theta(x) = \frac{1}{2} \left(x + (-1)^i \theta\right)^2, \quad i = 1, 2, \quad \theta > 0. \tag{2.37}$$

Similarly to D–NC, with D–NG we show in Appendix A that (2.36) also holds for the 2-node connected network, the symmetric W with $W_{12} = W_{21} = 1 - W_{11} = 1 - W_{22} = \frac{1}{2}\left(1 - 10^{-6}\right)$ (this W obeys Assumption 2.1), $\alpha_k = c/(k+1)$, and $c = \frac{1}{4} \times 10^{-6}$. The candidate functions are in (2.37), where, for fixed $k \geq 5$, $M > 0$, $\theta(k, M) = 8 \times 10^6\, k\, \sqrt{M}$.

We convey here the intuition why (2.36) holds for D–NG and D–NC, while the proof is in Section 2.8. Note that the solution to (2.1) with the f_i's in (2.37) is $x^\star = 0$, while $x_i^\star := \arg\min_{x \in \mathbb{R}} f_i(x) = (-1)^{i+1}\theta$, $i = 1, 2$. Making $x_1^\star$ and $x_2^\star$ to be far apart (by taking a large θ), problem (2.1) for D–NG and D–NC becomes "increasingly difficult." This is because the inputs to the disagreement dynamics (2.22) $(I - J)\nabla F(y(k-1)) = (I - J)y(k-1) - (-\theta, \theta)^\top$ are arbitrarily large, even when $y(k-1)$ is close the solution $y(k-1) \approx (0, 0)^\top$.

Finally, we consider what occurs if we drop Assumption 2.3 with Theorems 2.7 and 2.10. We show with D–NG and the above "hard" examples that $\|\widetilde{x}(k)\| \geq \frac{\sqrt{2}\, c\, \theta}{2\, k}$, $\forall k \geq 5$. Hence, $\|\widetilde{x}(k)\|$ is arbitrarily large by choosing θ large enough. (see Appendix A.) Similarly, with D–NC: $\|\widetilde{x}(k)\| \geq \frac{\alpha\, \theta \sqrt{2}}{4\, k^2}$, $\forall k \geq 10$. (see Appendix A.)

2.8 Technical Proofs

Subsection 2.8.1 proves Theorem 2.7; Subsection 2.8.2 proves the lower bound in (2.34) on the optimality gap of [2]; and Subsection 2.8.3 prove (2.36) for the D–NC method. The remaining proofs are in Appendix A.

2.8.1 Proof of Theorem 2.7

For notational simplicity, we let $d = 1$, but the proof extends to $d > 1$. We outline the main steps in the proof. First, we unwind the recursion (2.22) and calculate the underlying time varying system matrices. Second, we upper bound the norms of the time varying system matrices. Finally, we use these bounds and a summation argument to complete the proof of the Theorem.

Unwinding (2.22) **and calculating the system matrices**

Define the $2N \times 2N$ system matrices:

$$\Phi(k,t) := \Pi_{s=2}^{k-t+1} \begin{bmatrix} (1+\beta_{k-s})\widetilde{W} & -\beta_{k-s}\widetilde{W} \\ I & 0 \end{bmatrix}, \; k > t, \tag{2.38}$$

and $\Phi(k,k) = I$. Unwinding (2.22), the solution to (2.22) is:

$$\begin{bmatrix} \widetilde{x}(k) \\ \widetilde{x}(k-1) \end{bmatrix} = \sum_{t=0}^{k-1} \Phi(k,t+1)\, \alpha_t \begin{bmatrix} -(I-J)\nabla F(y(t)) \\ 0 \end{bmatrix}, \; k = 1,2,... \tag{2.39}$$

We now show the interesting structure of the matrix $\Phi(k,t)$ in (2.38) by decomposing it into the product of an orthonormal matrix U, a block-diagonal matrix, and $U^\top$. While U is independent of k and t, the block diagonal matrix depends on k and t, and has 2×2 diagonal blocks. Consider the matrix in (2.22) with $k-2 = t$, for a generic $t = -1, 0, 1, ...$ Using $\widetilde{W} = Q\widetilde{\Lambda}Q^\top$:

$$\begin{bmatrix} (1+\beta_t)\widetilde{W} & -\beta_t\widetilde{W} \\ I & 0 \end{bmatrix} = (Q \oplus Q)\, P \left(\oplus_{i=1}^{N} \Sigma_i(t)\right) P^\top (Q \oplus Q)^\top, \tag{2.40}$$

where P is the $2N \times 2N$ permutation matrix (e_i here is the i–th column of the $2N \times 2N$ identity matrix) $P = [e_1, e_{N+1}, e_2, e_{N+2}, ..., e_N, e_{2N}]^\top$, and $\Sigma_i(t)$ is a 2×2 matrix:

$$\Sigma_i(t) = \begin{bmatrix} (1+\beta_t)\lambda_i(\widetilde{W}) & -\beta_t\lambda_i(\widetilde{W}) \\ 1 & 0 \end{bmatrix}. \tag{2.41}$$

Using (2.40), and the fact that $(Q\oplus Q)P$ is orthonormal: $((Q\oplus Q)P)\cdot((Q\oplus Q)P)^\top = (Q\oplus Q)PP^\top(Q\oplus Q)^\top = (QQ^\top)\oplus(QQ^\top) = I$, we can express $\Phi(k,t)$ in (2.38) as:

$$\Phi(k,t) := (Q\oplus Q)P\left(\oplus_{i=1}^{N}\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)\right)P^\top(Q\oplus Q)^\top, \text{ for } k>t; \quad \Phi(k,k)=I. \tag{2.42}$$

Bounding the norm of $\Phi(k,t)$

As $(Q\oplus Q)P$ is orthonormal, $\Phi(k,t)$ has the same singular values as $\oplus_{i=1}^{N}\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)$, and so these two matrices also share the same spectral norm (maximal singular value.) Further, the matrix $\oplus_{i=1}^{N}\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)$ is block diagonal (with 2×2 blocks $\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)$), and so:

$$\|\Phi(k,t)\| = \max_{i=1,\ldots,N}\left\|\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)\right\|.$$

We proceed by calculating $\left\|\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)\right\|$. We distinguish two cases: $i=1$, and $i>1$.

Case $i=1$. As $\lambda_1(\widetilde{W})=0$, for all t, $\Sigma_1(t)=\Sigma_1$ is a constant matrix, with $[\Sigma_1]_{21}=1$, and the entries $(1,1)$, $(1,2)$ and $(2,2)$ of Σ_1 are zero. Note that $\|\Sigma_1\|=1$, and $(\Sigma_1)^s=0$, $s\geq 2$. Thus, as long as $k>t+1$, the product $\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)=0$, and so:

$$\left\|\Pi_{s=2}^{k-t+1}\Sigma_1(k-s)\right\| = \begin{cases} 1 & \text{if } k=t+1 \\ 0 & \text{if } k>t+1. \end{cases} \tag{2.43}$$

Case $i>1$. To simplify notation, let $\lambda_i := \lambda_i(\widetilde{W})$, and recall $\lambda_i\in(0,1)$; $\Sigma_i(t)$ is: $\Sigma_i(t)=\widehat{\Sigma}_i-\frac{3}{t+3}\Delta_i$, where: 1) $[\widehat{\Sigma}_i]_{11}=2\lambda_2$, $[\widehat{\Sigma}_i]_{12}=-\lambda_i$, $[\widehat{\Sigma}_i]_{21}=1$, and $[\widehat{\Sigma}_i]_{22}=0$; and 2) $[\Delta_i]_{11}=-[\Delta_i]_{12}=\lambda_i$, and $[\Delta_i]_{21}=[\Delta_i]_{22}=0$. ; $\widehat{\Sigma}_i$ is diagonalizable, with $\widehat{\Sigma}_i=\widehat{\mathcal{Q}_i}\widehat{\mathcal{D}}_i\widehat{\mathcal{Q}_i}^{-1}$, and:

$$\widehat{\mathcal{Q}_i}=\begin{bmatrix}\lambda_i+\mathbf{j}\sqrt{\lambda_i(1-\lambda_i)} & \lambda_i-\mathbf{j}\sqrt{\lambda_i(1-\lambda_i)}\\ 1 & 1\end{bmatrix},\ \widehat{\mathcal{D}}_i=\begin{bmatrix}\lambda_i+\mathbf{j}\sqrt{\lambda_i(1-\lambda_i)} & 0\\ 0 & \lambda_i-\mathbf{j}\sqrt{\lambda_i(1-\lambda_i)}\end{bmatrix}.$$

(Note that the matrices $\widehat{\mathcal{Q}}_i$ and $\widehat{\mathcal{D}}_i$ are complex.) Denote by $\mathcal{D}_i(t)=\widehat{\mathcal{D}}_i-\frac{3}{t+3}\widehat{\mathcal{Q}_i}^{-1}\Delta_i\widehat{\mathcal{Q}_i}$. Then, $\Sigma_i(t)=\widehat{\mathcal{Q}}_i\left(\widehat{\mathcal{D}}_i-\frac{3}{t+3}\widehat{\mathcal{Q}}_i^{-1}\Delta_i\widehat{\mathcal{Q}}_i\right)\widehat{\mathcal{Q}}_i^{-1}=\widehat{\mathcal{Q}_i}\mathcal{D}_i(t)\widehat{\mathcal{Q}_i}^{-1}$. By the sub-multiplicative property of norms, and using $\left\|\widehat{\mathcal{Q}_i}\right\|\leq\sqrt{2}\left\|\widehat{\mathcal{Q}_i}\right\|_\infty=2\sqrt{2}$, $\left\|\widehat{\mathcal{Q}_i}^{-1}\right\|\leq\sqrt{2}\left\|\widehat{\mathcal{Q}_i}^{-1}\right\|_\infty=\frac{2\sqrt{2}}{\sqrt{\lambda_i(1-\lambda_i)}}$:

$$\|\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)\|\leq\frac{8}{\sqrt{\lambda_i(1-\lambda_i)}}\,\Pi_{s=2}^{k-t+1}\|\mathcal{D}_i(k-s)\|. \tag{2.44}$$

It remains to upper bound $\|\mathcal{D}_i(t)\|$, for all $t = -1, 0, 1, ...$ We will show that

$$\|\mathcal{D}_i(t)\| \leq \sqrt{\lambda_i}, \ \ \forall t = -1, 0, 1, ... \tag{2.45}$$

Denote by $a_t = \frac{3}{t+3}$, $t = 0, 1, ...$, and $a_{-1} = 1$. After some algebra, the entries of $\mathcal{D}_i(t)$ are: $[\mathcal{D}_i(t)]_{11} = ([\mathcal{D}_i(t)]_{22})^H = \frac{1}{2}(2 - a_t)(\lambda_i + \mathbf{j}\sqrt{\lambda_i(1-\lambda_i)})$, $[\mathcal{D}_i(t)]_{12} = ([\mathcal{D}_i(t)]_{21})^H = a_t(\lambda_i + \mathbf{j}\sqrt{\lambda_i(1-\lambda_i)})$, which gives: $[\mathcal{D}_i(t)^H\mathcal{D}_i(t)]_{11} = [\mathcal{D}_i(t)^H\mathcal{D}_i(t)]_{22} = \frac{a_t^2+(2-a_t)^2}{4}\lambda_i$, and $[\mathcal{D}_i(t)^H\mathcal{D}_i(t)]_{12} = \Big([\mathcal{D}_i(t)^H\mathcal{D}_i(t)]_{21}\Big)^H = \frac{a_t(2-a_t)}{2}\left(2\lambda_i^2 - \lambda_i - 2\mathbf{j}\lambda_i\sqrt{\lambda_i(1-\lambda_i)}\right)$. Next, very interestingly:

$$\|\mathcal{D}_i{}^H(t)\mathcal{D}_i(t)\|_1 = \left\|\left[\mathcal{D}_i{}^H(t)\mathcal{D}_i(t)\right]_{11}\right\| + \left\|\left[\mathcal{D}_i{}^H(t)\mathcal{D}_i(t)\right]_{12}\right\| = \frac{1}{4}(a_t^2 + (2-a_t)^2)\lambda_i + \frac{1}{2}a_t(2-a_t)\lambda_i, = \lambda_i.$$

for any $a_t \in [0, 2]$, which is the case here because $a_t = 3/(t+3)$, $t = 0, 1, ...$, and $a_{-1} = 0$. Thus, as $\|A\| \leq \|A\|_1$ for a Hermitean matrix A: $\|\mathcal{D}_i(t)\| = \sqrt{\|\mathcal{D}_i{}^H(t)\mathcal{D}_i(t)\|} \leq \sqrt{\|\mathcal{D}_i{}^H(t)\mathcal{D}_i(t)\|_1} = \sqrt{\lambda_i}$. Applying the last equation and (2.45) to (2.44), we get, for $i \neq 1$: $\|\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)\| \leq \frac{8}{\sqrt{\lambda_i(1-\lambda_i)}}\left(\sqrt{\lambda_i}\right)^{k-t}$, $k \geq t+1$. Combine the latter with (2.43), and use $\|\Phi(k,t)\| = \max_{i=1,...,N}\|\Pi_{s=2}^{k-t+1}\Sigma_i(k-s)\|$, Assumption 2.1 (b) and $\lambda_N(\widetilde{W}) = \mu(W)$, to obtain:

$$\|\Phi(k,t)\| \leq \frac{8\left(\sqrt{\mu(W)}\right)^{k-t}}{\min_{i\in\{2,N\}}\sqrt{\lambda_i(\widetilde{W})(1-\lambda_i(\widetilde{W}))}} \leq \frac{8}{\sqrt{\kappa(1-\mu(W))}}\left(\sqrt{\mu(W)}\right)^{k-t}, \ k \geq t. \tag{2.46}$$

Summation

We apply (2.46) to (2.39). Using the sub-multiplicative and sub-additive properties of norms, expression $\alpha_t = c/(t+1)$, and the inequalities $\|\widetilde{x}(k)\| \leq \left\|(\widetilde{x}(k)^\top, \widetilde{x}(k-1)^\top)^\top\right\|$, $\left\|(-(I-J)\nabla F(y(t))^\top, 0^\top)^\top\right\| \leq \sqrt{N}\,G$:

$$\|\widetilde{x}(k)\| \leq \frac{8\sqrt{N}\,c\,G}{\sqrt{\kappa(1-\mu(W))}}\sum_{t=0}^{k-1}\left(\sqrt{\mu(W)}\right)^{k-(t+1)}\frac{1}{(t+1)}. \tag{2.47}$$

We now denote by $r := \sqrt{\mu(W)} \in (0,1)$. To complete the proof of the Lemma, we upper bound the sum $\sum_{t=0}^{k-1} r^{k-(t+1)}\frac{1}{(t+1)}$ by splitting it into two sums. With the first sum, t runs from zero to $\lceil k/2\rceil$, while with the second sum, t runs from $\lceil k/2\rceil + 1$ to k :

$$\sum_{t=0}^{k-1}\frac{r^{k-(t+1)}}{t+1} = \left(r^{k-1} + r^{k-2}\frac{1}{2} + ... + r^{\lceil k/2\rceil}\frac{1}{\lceil k/2\rceil}\right) + \left(r^{k-(\lceil k/2\rceil+1)}\frac{1}{\lceil k/2\rceil+1} + ... + \frac{1}{k}\right)$$

$$
\begin{aligned}
&\leq r^{k/2}\left(1+\frac{1}{2}+...+\frac{1}{k/2}+\frac{1}{(k+1)/2}\right)+\frac{1}{(k/2)}\left(1+r+...+r^k\right)\\
&\leq r^{k/2}\left(\log(1+k/2)+2\right)+\frac{2}{k}\frac{1}{1-r} \qquad (2.48)\\
&= 2\left\{r^{k/2}\log(1+k/2)(k/2)\right\}\frac{1}{k}+\left\{4r^{k/2}(k/2)\right\}\frac{1}{k}+\frac{2}{k}\frac{1}{1-r} \qquad (2.49)\\
&\leq 2\sup_{z\geq 1/2}\left\{r^z\log(1+z)z\right\}\frac{1}{k}+4\sup_{z\geq 1/2}\left\{r^z z\right\}\frac{1}{k}+\frac{2}{k}\frac{1}{1-r} \qquad (2.50)\\
&\leq \left(2\mathcal{B}(r)+\frac{4}{e(-\log r)}+\frac{2}{1-r}\right)\frac{1}{k}\leq\left(2\mathcal{B}(r)+\frac{7}{1-r^2}\right)\frac{1}{k}. \qquad (2.51)
\end{aligned}
$$

Inequality (2.48) uses the inequality $1+\frac{1}{2}+...+\frac{1}{t}\leq\log t+1,\ t=1,2,...$, and $1+r+...+r^k\leq\frac{1}{1-r}$; (2.49) multiplies and divides the first summand on the right hand side of (2.48) by $k/2$; (2.50) uses $r^{k/2}\log(1+k/2)(k/2)\leq\sup_{z\geq 1/2}r^z\log(1+z)z$, for all $k=1,2,...$, and a similar bound for the second summand in (2.49); the left inequality in (2.51) uses $\mathcal{B}(r):=\sup_{z\geq 1/2}r^z\log(1+z)z$ and $\sup_{z\geq 1/2}r^z\,z\leq\frac{1}{e\,(-\log r)}$ (note that $r^z\,z$ is convex in z; we take the derivative of $r^z\,z$ with respect to z and set it to zero); and the right inequality in (2.51) uses $-1/\log r\leq 1/(1-r)$, $\forall r\in[0,1)$; $1/(1-r)\leq 2/(1-r^2)$, $\forall r\in[0,1)$, and $e=2.71...$ Applying the last to (2.47), and using the C_{cons} in (2.21), Theorem 2.7 for $\|\widetilde{x}(k)\|$ follows. Then, as $\widetilde{y}(k)=\widetilde{x}(k)+\frac{k-1}{k+2}(\widetilde{x}(k)-\widetilde{x}(k-1))$, we have that $\|\widetilde{y}(k)\|\leq 2\|\widetilde{x}(k)\|+\|\widetilde{x}(k-1)\|$. Further, by Theorem 2.7: $\|\widetilde{x}(k-1)\|\leq c\sqrt{N}GC_{\text{cons}}\frac{1}{k-1}\frac{k}{k}\leq 2c\sqrt{N}GC_{\text{cons}}\frac{1}{k}$, $k\geq 2$, and $\|\widetilde{x}(0)\|=0$ (by assumption). Thus, $\|\widetilde{x}(k-1)\|\leq 2c\sqrt{N}GC_{\text{cons}}\frac{1}{k}$, $\forall k\geq 1$. Thus, $\|\widetilde{y}(k)\|\leq 2\|\widetilde{x}(k)\|+\|\widetilde{x}(k-1)\|\leq 4c\sqrt{N}GC_{\text{cons}}\frac{1}{k}$, $\forall k\geq 1$.

2.8.2 Proof of the worst-case lower bound for standard distributed gradient method

Consider the f_i's in (2.35), the initialization $x_i(0)=(1,0)^\top$, $i=1,2$, and $W_{12}=W_{21}=1-W_{11}=1-W_{22}=w=1/8$, as we set in Subsection 2.7.1. We divide the proof in four steps. First, we prove certain properties of (2.1) and the f_i's in (2.35); second, we solve for the state $x(k)=(x_1(k)^\top,x_2(k)^\top)^\top$ with the algorithm in [2]; third, we upper bound $\|x(k)\|$; finally, we use the latter bound to derive the $\Omega(1/k^{2/3})$ worst-case optimality gap.

Step 1: Properties of the f_i^θ's

Consider the f_i^θ's in (2.35) for a fixed $\theta\in[0,1]$. The solution to (2.1), with $f(x)=f_1^\theta(x)+f_2^\theta(x)$, is $x^\star=(0,0)^\top$, and the corresponding optimal value is $f^\star=\theta+1$. Further, the f_i^θ's belong to the class $\mathcal{F}(L=\sqrt{2},G=10)$. (Proof is in the Supplementary material.)

Step 2: Solving for $x(k)$ with the algorithm in [2]

Now, consider the algorithm in [2], and consider $x_i(k)$–the solution estimate at node i and time k. Denote by $x^l(k) = (x_1^{(l)}(k), x_2^{(l)}(k))^\top$–the vector with the l-th coordinate of the estimate of both nodes, $l = 1, 2$; and $d^l(k) = \left(\frac{\partial f_1(x_1(k))}{\partial x^{(l)}}, \frac{\partial f_2(x_2(k))}{\partial x^{(l)}}\right)^\top$, $l = 1, 2$. Then, the update rule of [2] is, for the f_1^θ, f_2^θ in (2.35):

$$x^l(k) = Wx^l(k-1) - \alpha_{k-1}d^l(k-1), \;\; k = 1, 2, ..., \;\; l = 1, 2. \tag{2.52}$$

Recall the "hard" initialization $x^{\mathrm{I}}(0) = (1, 1)^\top$, $x^{\mathrm{II}}(0) = (0, 0)^\top$. Under this initialization:

$$x_i(k) \in \mathcal{R}_i := \left\{x \in \mathbb{R}^2 : \; \theta(x^{(1)} + (-1)^i)^2 + (x^{(2)} + (-1)^i)^2 \leq \overline{\chi}^2\right\}, \tag{2.53}$$

for all k, for both nodes $i = 1, 2$ (proof in the Supplementary material.) Note that $\mathcal{R}_i$ is the region where the f_i^θ in (2.35) is quadratic. Thus, evaluating ∇f_i^θ's in the quadratic region:

$$x^l(k) = \left(W - \alpha_{k-1}\kappa^l I\right) x^{\mathrm{I}}(k-1) - \alpha_{k-1}\kappa^l \left(-1, 1\right)^\top, \tag{2.54}$$

$l = 1, 2$, where $\kappa^{\mathrm{I}} = \theta$ and $\kappa^{\mathrm{II}} = 1$. We now evaluate $\sum_{i=1}^2 \left(f(x_i(k)) - f^\star\right)$, $f(x) = f_1^\theta(x) + f_2^\theta(x)$. Because $x_i(k) \in \mathcal{R}_i$, $i = 1, 2$, verify, using (2.35), and $f^\star = 1 + \theta$, that:

$$\sum_{i=1}^2 \left(f(x_i(k)) - f^\star\right) = \theta\|x^{\mathrm{I}}(k)\|^2 + \|x^{\mathrm{II}}(k)\|^2. \tag{2.55}$$

By unwinding (2.54), and using $x^{\mathrm{I}}(0) = (1, 1)^\top$, $x^{\mathrm{II}}(0) = (0, 0)^\top$:

$$\begin{aligned} x^{\mathrm{I}}(k) &= \left(W - \alpha_{k-1}\theta I\right)\left(W - \alpha_{k-2}\theta I\right)...\left(W - \alpha_0\theta I\right)(1, 1)^\top \\ &+ \theta\left(\sum_{t=0}^{k-2}(W - \alpha_{k-1}\theta I)(W - \alpha_{k-2}\theta I)...(W - \alpha_{t+1}\theta I)\alpha_t + \alpha_{k-1}I\right)(1, -1)^\top \\ x^{\mathrm{II}}(k) &= \left(\sum_{t=0}^{k-2}(W - \alpha_{k-1}I)(W - \alpha_{k-2}I)...(W - \alpha_{t+1}I)\alpha_t + \alpha_{k-1}I\right)(1, -1)^\top. \end{aligned}$$

Consider the eigenvalue decomposition $W = Q\Lambda Q^\top$, where $Q = [q_1, q_2]$, $q_1 = \frac{1}{\sqrt{2}}(-1, 1)^\top$, $q_2 = \frac{1}{\sqrt{2}}(1, 1)^\top$, and Λ is diagonal with the eigenvalues $\Lambda_{11} = \lambda_1 = 1 - 2w = 3/4$, $\Lambda_{22} = \lambda_2 = 1$. The matrix $W - \alpha_{k-1}\theta I$ decomposes as $W - \alpha_{k-1}\theta I = Q(\Lambda - \alpha_{k-1}\theta I)Q^\top$; likewise, $W - \alpha_{k-1}I = Q(\Lambda - \alpha_{k-1}I)Q^\top$. Then, $(W - \alpha_{k-1}\theta I)(W - \alpha_{k-2}\theta I)...(W - \alpha_{t+1}\theta I) = Q(\Lambda - \alpha_{k-1}\theta I)...(\Lambda - \alpha_{t+1}\theta I)Q^\top$, and $(W - \alpha_{k-1}I)...(W - \alpha_{t+1}I) = Q(\Lambda - \alpha_{k-1}I)...(\Lambda - \alpha_{t+1}I)Q^\top$. Using these decompositions, and

the orthogonality: $q_1^\top (1,1)^\top = 0$, and $q_2^\top (-1,1)^\top = 0$:

$$\begin{aligned} x^{\mathrm{I}}(k) &= (1-\alpha_{k-1}\theta)\,(1-\alpha_{k-2}\theta)\,...\,(1-\alpha_0\theta)\,(1,1)^\top \\ &+ \theta(1,-1)^\top \left(\sum_{t=0}^{k-2} (\lambda_1 - \alpha_{k-1}\theta)(\lambda_1 - \alpha_{k-2}\theta)...(\lambda_1 - \alpha_{t+1}\theta)\alpha_t + \alpha_{k-1} \right) \end{aligned} \tag{2.56}$$

$$x^{\mathrm{II}}(k) = (1,-1)^\top \left(\sum_{t=0}^{k-2} (\lambda_1 - \alpha_{k-1})(\lambda_1 - \alpha_{k-2})...(\lambda_1 - \alpha_{t+1})\alpha_t + \alpha_{k-1} \right). \tag{2.57}$$

Step 3: Upper bounding $\|x(k)\|$

Note that $\lambda_1 - \alpha_{k-1}\theta = 3/4 - \frac{c\theta}{k^\tau} \geq 1/4$, for all k, τ, c. Also, $\lambda_1 - \alpha_{k-1}\theta \leq \lambda_1 = 3/4$, for all k, τ, c. Similarly, we can show $1 - \alpha_{k-1}\theta \in [1/2, 1]$. (Note that the terms $(1-\alpha_{k-1}\theta)...(1-\alpha_0\theta)$, $(\lambda_1 - \alpha_{k-1}\theta)...(\lambda_1 - \alpha_{t+1}\theta)$, and $(\lambda_1 - \alpha_{k-1})...(\lambda_1 - \alpha_{t+1})$ are then nonnegative, $\forall t$.) Thus: $\|x^{\mathrm{I}}(k)\| \geq (1-\alpha_{k-1}\theta)\,(1-\alpha_{k-2}\theta)\,...\,(1-\alpha_0\theta)$. Set $\theta = \theta_k = 1/(s_k(\tau)) \leq 1$, where $s_k(\tau) := \sum_{t=0}^{k-1}(t+1)^{-\tau}$; use $(1-a_1)(1-a_2)...(1-a_n) \geq 1 - (a_1 + a_2 + ... + a_n)$, $a_i \in [0,1)$, $\forall i$; and $\alpha_k = \frac{c}{(k+1)^\tau}$. We obtain: $\|x^{\mathrm{I}}(k)\| \geq 1 - c\,\theta_k\,s_k(\tau)$, and so: $\theta_k \|x^{\mathrm{I}}(k)\|^2 \geq \frac{(1-c_{\max})^2}{s_k(\tau)}$, where we denote $c_{\min} := c_0$ and $c_{\max} := 1/(2L) = 1/(2\sqrt{2})$. Further, from (2.57): $\|x^{\mathrm{II}}(k)\|^2 \geq \alpha_{k-1}^2 \geq \frac{c_{\min}^2}{k^{2\tau}}$, and we obtain:

$$\theta_k \,\|x^{\mathrm{I}}(k)\|^2 + \|x^{\mathrm{II}}(k)\|^2 \geq \frac{(1-c_{\max})^2}{s_k(\tau)} + \frac{c_{\min}^2}{k^{2\tau}}. \tag{2.58}$$

Step 4: Upper bounding the optimality gap from (2.58)

From (2.58), and using (2.55):

$$\max_{i=1,2} (f(x_i(k)) - f^\star) \geq \frac{1}{2}\sum_{i=1}^{2} (f(x_i(k)) - f^\star) \geq \frac{(1-c_{\max})^2}{2s_k(\tau)} + \frac{c_{\min}^2}{2\,k^{2\tau}} =: e_k(\tau), \tag{2.59}$$

$\forall k \geq 1$, $\forall \tau \geq 0$. We further upper bound the right hand side in (2.59) by taking the infimum of $e_k(\tau)$ over $\tau \in [0, \infty)$; we split the interval $[0,\infty)$ into $[0, 3/4]$; $[3/4, 1]$, and $[1, \infty)$, so that

$$\inf_{[0,\infty)} e_k(\tau) \geq \min \left\{ \inf_{[0,3/4]} e_k(\tau),\ \inf_{[3/4,1]} e_k(\tau),\ \inf_{[1,\infty)} e_k(\tau) \right\}. \tag{2.60}$$

It is easy to prove that: 1) $\inf_{[0,3/4)} e_k(\tau) = \Omega(1/k^{2/3})$; 2) using $s_k(\tau) \leq 3(\log k)(k+1)^{1-\tau}$, $\forall k \geq 3$, $\forall \tau \in [0,1]$, that $\inf_{[3/4,1]} e_k(\tau) = \Omega\left(\frac{1}{(\log k)k^{1/4}}\right)$; and 3) $\inf_{[1,\infty)} e_k(\tau) = \Omega\left(\frac{1}{\log k}\right)$. (see the Supplementary material.) Combining the latter bounds with (2.60) completes the proof of (2.34).

2.8.3 Relaxing bounded gradients: Proof of (2.36) for D–NC

We prove (2.36) for D–NC while the proof of D–NG is similar and is in the Supplementary material. Fix arbitrary $\theta > 0$ and take the f_i's in (2.37). From (2.13)–(2.14), evaluating the ∇f_i's:

$$x(k) = (1-\alpha)W^{\tau_x(k)}y(k-1) + \alpha\,\theta\,W^{\tau_x(k)}(1,-1)^\top, \quad y(k) = W^{\tau_y(k)}\left(x(k) + \beta_{k-1}(x(k)-x(k-1))\right), \tag{2.61}$$

for $k = 1, 2, ...$ We take the initialization at the solution $x(0) = y(0) = (0,0)^\top$. Consider the eigenvalue decomposition $W = Q\Lambda Q^\top$, with $Q = [q_1, q_2]$, $q_1 = \frac{1}{\sqrt{2}}(1,-1)^\top$, $q_2 = \frac{1}{\sqrt{2}}(1,1)^\top$, and Λ is diagonal with $\Lambda_{11} = \lambda_1$, $\Lambda_{22} = \lambda_2 = 1$. Define $z(k) = Q^\top x(k)$ and $w(k) = Q^\top y(k)$. Multiplying (2.61) from the left by $Q^\top$, and using $Q^\top(1,-1)^\top = (\sqrt{2},0)^\top$:

$$\begin{aligned} z(k) &= (1-\alpha)\Lambda^{\tau_x(k)}w(k-1) + \alpha\,\theta\Lambda^{\tau_x(k)}(\sqrt{2},0)^\top, \\ w(k) &= \Lambda^{\tau_y(k)}\left[z(k) + \beta_{k-1}(z(k)-z(k-1))\right], \end{aligned} \tag{2.62}$$

$k = 1, 2, ...$, and $z(0) = w(0) = (0,0)^\top$. Next, note that

$$\max_{i=1,2}(f(x_i(k)) - f^\star) \geq \frac{1}{2}\sum_{i=1}^{2}(f(x_i(k)) - f^\star) = \frac{\|x(k)\|^2}{2} = \frac{\|z(k)\|^2}{2} \geq \frac{(z^{(1)}(k))^2}{2}. \tag{2.63}$$

Further, from (2.62) for the first coordinate $z^{(1)}(k), w^{(1)}(k)$, recalling that $\mu := \lambda_1$:

$$\|z^{(1)}(k)\| \leq \mu^{\tau_x(k)}\|w^{(1)}(k-1)\| + \sqrt{2}\,\alpha\,\theta\,\mu^{\tau_x(k)}, \;\; \|w^{(1)}(k)\| \leq \mu^{\tau_y(k)}\left(2\|z^{(1)}(k)\| + \|z^{(1)}(k-1)\|\right), \tag{2.64}$$

$k = 1, 2, ...$ Note that (2.64) is analogous to (2.31)–(2.32) with the identification $\widetilde{x}(k) \equiv z^{(1)}(k)$, $\widetilde{y}(k) \equiv w^{(1)}(k)$, $\sqrt{N}G \equiv \sqrt{2}\theta$; hence, analogously to the proof of Theorem 2.10, from (2.64): $\|w^{(1)}(k-1)\| \leq \frac{2\sqrt{2}\,\alpha\,\theta}{(k-1)^2}$, $k = 2, 3...$Using the latter, (2.62), and $\frac{1}{k^2} \geq \mu^{\tau_x(k)} \geq \frac{1}{e\,k^2}$ (see (2.11)): $\|z^{(1)}(k)\| \geq \alpha\,\theta\sqrt{2}\mu^{\tau_x(k)} - \mu^{\tau_x(k)}\|w^{(1)}(k-1)\| \geq \frac{\alpha\theta\sqrt{2}}{e\,k^2}\left(1 - \frac{2\,e}{(k-1)^2}\right) \geq \frac{\alpha\,\theta\sqrt{2}}{4\,k^2} > 0, \;\; \forall k \geq 10$. Thus, from (2.63) and the latter inequality, $\max_{i=1,2}(f(x_i(k)) - f^\star) \geq \frac{\alpha^2\theta^2}{16\,k^4}$, which is, for $\alpha = 1/(2L) = 1/2$, greater or equal M for $\theta = \theta(k, M) = 8\sqrt{M}\,k^2$.

2.9 Simulations

We compare the proposed D–NG and D–NC algorithms with [2, 12, 29] on the logistic loss. Simulations confirm the increased convergence rates of D–NG and D–NC with respect to [2, 12] and show a comparable performance with respect to [29]. More precisely, D–NG achieves an accuracy ϵ faster than [2, 12] for

all ϵ, while D–NC is faster than [2, 12] at least for $\epsilon \leq 10^{-2}$. With respect to [29], D–NG is faster for lower accuracies (ϵ in the range 10^{-1} to $10^{-4} - 10^{-5}$), while [29] becomes faster for high accuracies ($10^{-4} - 10^{-5}$ and finer); D–NC performs slower than [29].

Simulation setup

We consider distributed learning via the logistic loss; see, e.g., [78] for further details. Nodes minimize the logistic loss: $f(x) = \sum_{i=1}^{N} f_i(x) = \sum_{i=1}^{N} \log\left(1 + e^{-b_i(a_i^\top x_1 + x_0)}\right)$, where $x = (x_1^\top, x_2^\top)^\top$, $a_i \in \mathbb{R}^2$ is the node i's feature vector, and $b_i \in \{-1, +1\}$ is its class label. The functions $f_i : \mathbb{R}^d \mapsto \mathbb{R}$, $d = 3$, satisfy Assumptions 2.2 and 2.3. The Hessian $\nabla^2 f(x) = \sum_{i=1}^{N} \frac{e^{-c_i^\top x}}{(1+e^{-c_i^\top x})^2} c_i c_i^\top$, where $c_i = (b_i a_i^\top, b_i)^\top \in \mathbb{R}^3$. A Lipschitz constant L should satisfy $\|\nabla^2 f(x)\| \leq NL, \forall x \in \mathbb{R}^d$. Note that $\nabla^2 f(x) \preceq \frac{1}{4} \sum_{i=1}^{N} c_i c_i^\top$, because $\frac{e^{-c_i^\top x}}{(1+e^{-c_i^\top x})^2} \leq 1/4$, $\forall x$. We thus choose $L = \frac{1}{4N} \left\| \sum_{i=1}^{N} c_i c_i^\top \right\| \approx 0.3053$. We generate a_i independently over i; each entry is drawn from the standard normal distribution. We generate the "true" vector $x^\star = (x_1^{\star\top}, x_0^\star)^\top$ by drawing its entries independently from the standard normal distribution. The labels are $b_i = \text{sign}\left(x^\star{}_1^\top a_i + x^\star{}_0 + \epsilon_i\right)$, where the ϵ_i's are drawn independently from a normal distribution with zero mean and variance 3. The network is a geometric network: nodes are placed uniformly randomly on a unit square and the nodes whose distance is less than a radius are connected by an edge. There are $N = 100$ nodes, and the relative degree $\left(= \frac{\text{number of links}}{N(N-1)/2}\right) \approx 10\%$. We initialize all nodes by $x_i(0) = 0$ (and $y_i(0) = 0$ with D–NG, D–NC, and [29]). With all algorithms except D–NG, we use the Metropolis weights W [65]; with D–NG, we use $W' = \frac{1+\kappa}{2} I + \frac{1-\kappa}{2} W$, with $\kappa = 0.1$. The step-size α_k is: $\alpha_k = 1/(k+1)$, with D–NG; $\alpha = 1/(2L)$ and $1/L$, with D–NC[6]; $1/L$, with [29] (both the 1st and 2nd algorithm variants – see Subsection 2.7.1); and $1/(k+1)^{1/2}$, with [2] and [12]. [7] We simulate the normalized (average) error $\frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i) - f^\star}{f(x_i(0)) - f^\star}$ versus the total number of communications at all nodes ($= N\mathcal{K}$.)

Results

Figure 2.2 (left) compares D–NG, D–NC (with step-sizes $\alpha = 1/(2L)$ and $1/L$), [2, 12], [29] (both 1st and 2nd variant with $\alpha = 1/L$.) We can see that D–NG converges faster than other methods for accuracies ϵ in the range 10^{-1} to $3 \cdot 10^{-5}$. For example, for $\epsilon = 10^{-2}$, D–NG requires about 10^4 transmissions; [29] (2nd variant) $\approx 3.16 \cdot 10^4$; D–NC ($\alpha = 1/L$) $\approx 4.65 \cdot 10^4$, and D–NC with $\alpha = 1/(2L) \approx 1.1 \cdot 10^5$; and [29] (1st variant), [2], and [12] – at least $\approx 1.3 \cdot 10^5$. For high accuracies, $2 \cdot 10^{-5}$ and finer, [29] (2nd variant) becomes faster than D–NG. Finally, [29] (2nd) converges faster than D–NC, while [29] (1st) is slower than

[6] Our theoretical analysis allows for $\alpha \leq 1/(2L)$, but we also simulate D–NG with $\alpha = 1/L$.

[7] With [2, 12], $\alpha_k = 1/(k+1)^p$ and $p = 1/2$, gave the best simulation performance among the choices $p \in \{1/3, 1/2, 1\}$.

D–NC.

Further comparisons of D–NG and D–NC: Huber loss

We provide an additional experiment to further compare the D–NG and D–NC methods. We show that the relative performance of D–NC with respect to D–NG improves when the instance of (2.1) becomes easier (in the sense explained below.) We consider a $N = 20$-node geometric network with $\frac{\text{number of links}}{N(N-1)/2} \approx 32\%$ and Huber losses $f_i : \mathbb{R} \to \mathbb{R}$, $f_i(x) = \frac{1}{2}\|x - a_i\|^2$ if $\|x - a_i\| \leq 1$, and $f_i(x) = \|x - a_i\| - 1/2$, else, with $a_i \in \mathbb{R}$. We divide the set of nodes in two groups. For the first group, $i = 1, ..., 6$, we generate the a_i's as $a_i = \theta + \nu_i$, where $\theta > 0$ is the "signal" and ν_i is the uniform noise on $[-0.1\theta,\, 0.1\theta]$. For the second group, $i = 7, ..., 20$, we set $a_i = -\theta + \nu_i$, with the ν_i's from the same uniform distribution. Note that any $x_1^\star \in \arg\min_{x\in\mathbb{R}} \sum_{i=1}^{6} f_i(x)$ is in $[0.9\theta,\, 1.1\theta]$, while any $x_2^\star \in \arg\min_{x\in\mathbb{R}} \sum_{i=7}^{20} f_i(x)$ lies in $[-1.1\theta,\, -0.9\theta]$. Intuitively, by making $\theta > 0$ large, we increase the problem difficulty. For a small θ, we are in the "easy problem" regime, because the solutions $x_1^\star$ and $x_2^\star$ of the two nodes' groups are close; for a large θ, we are in the "difficult problem" regime. Figure 2.2 (right) plots the normalized average error versus $N\mathcal{K}$ for $\theta \in \{0.01; 10; 1000\}$ for D–NG with $\alpha_k = 1/(k+1)$, D–NC with $\alpha = 1/L$, while both algorithms are initialized by $x_i(0) = y_i(0) = 0$. We can see that, with D–NC, the decrease of θ makes the convergence faster, as expected. (With D–NG, it is not a clear "monotonic" behavior.) Also, as θ decreases ("easier problem"), the performance of D–NC relative do D–NG improves. For $\theta = 0.01$, D–NG is initially better, but the curves of D–NG and D–NC intersect at the value about $4 \cdot 10^{-3}$, while for $\theta = 1000$, D–NG is better for all accuracies as fine as (at least) 10^{-7}.

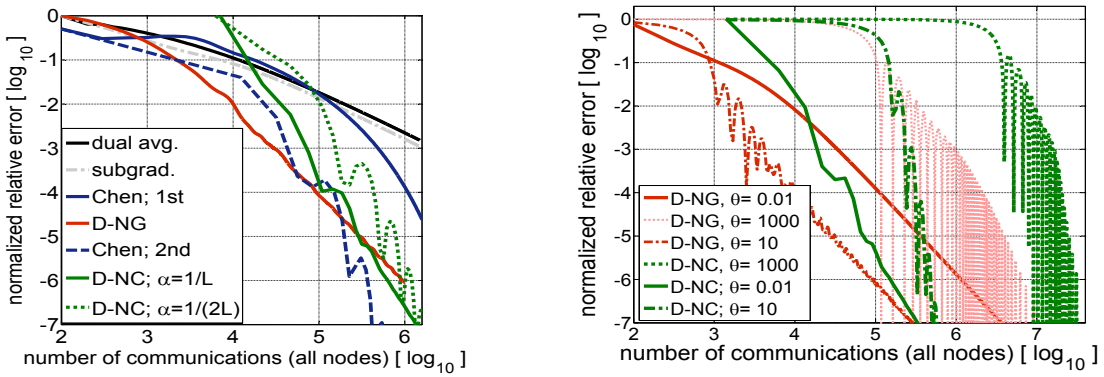

Figure 2.2: Normalized (average) relative error $\frac{1}{N}\sum_{i=1}^{N} \frac{f(x_i)-f^\star}{f(x_i(0))-f^\star}$ versus the number of communications (all nodes) $N\mathcal{K}$; Left: Logistic loss; Right: Huber loss.

2.10 Conclusion

We proposed fast distributed Nesterov-like gradient algorithms to solve optimization problem (1.1). Existing literature has presented distributed gradient based algorithms to solve this problem and has studied their convergence rates, for a class of convex, non-differentiable f_i's, with bounded gradients. In this chapter, we asked whether faster convergence rates than the rates established in the literature can be achieved on a more structured class of f_i's – convex, with Lipschitz continuous gradient (with constant L) and bounded gradient. Building from the centralized Nesterov gradient method, we answer affirmatively this question by proposing two distributed gradient algorithms. Our algorithm D–NG achieves the rates $O\left(\frac{1}{(1-\mu(W))^{3+\xi}}\frac{\log \mathcal{K}}{\mathcal{K}}\right)$ and $O\left(\frac{1}{(1-\mu(W))^{3+\xi}}\frac{\log k}{k}\right)$, when the global knowledge of the gradient Lipschitz constant L and the network spectral gap $1-\mu(W)$ is not available before the algorithm run. The rates, for the optimized step size, improve to $O\left(\frac{1}{(1-\mu(W))^{1+\xi}}\frac{\log \mathcal{K}}{\mathcal{K}}\right)$ and $O\left(\frac{1}{(1-\mu(W))^{1+\xi}}\frac{\log k}{k}\right)$, when L and $\mu(W)$ are available before the run. Our algorithm D–NC operates only if L and $\mu(W)$ are available and achieves rates $O\left(\frac{1}{(1-\mu(W))^2}\frac{1}{\mathcal{K}^{2-\xi}}\right)$ and $O\left(\frac{1}{k^2}\right)$. We also showed that our methods achieve strictly faster rates than the method in [2]. Simulations illustrate the performance of the proposed methods.

Chapter 3

Distributed Nesterov-like Gradient Methods: Random Networks

3.1 Introduction

In Chapter 2, we presented our D–NG and D–NC distributed methods for static networks. In many applications, it is relevant to account for *randomness* in the underlying network. Randomness arises when inter-node links fail as with random packet dropouts in wireless sensor networks, or when communication protocols are random like with the gossip protocol [10]. In this Chapter, we propose modified D–NG and D–NC algorithms, referred to as mD–NG and mD–NC, respectively, and establish their convergence rate guarantees on *random networks*.

We model the network by a sequence of random independent, identically distributed (i.i.d.) weight matrices $W(k)$ drawn from a set of symmetric, stochastic matrices with positive diagonals, and we assume that the network is connected on average[1]. We establish the convergence rates of the expected optimality gap in the cost function (at any node i)[2] of mD-NG and mD-NC, in terms of the number of per node gradient evaluations k and the number of per-node communications $\mathcal{K}$, when the functions f_i are convex and differentiable, with Lipschitz continuous and bounded gradients. We show that the modified methods achieve *in expectation* the same rates that the methods in Chapter 2 achieve on static networks, namely: mD–NG converges at rates $O(\log k/k)$ and $O(\log \mathcal{K}/\mathcal{K})$, while mD–NC has rates $O(1/k^2)$ and $O(1/\mathcal{K}^{2-\xi})$, where ξ is an arbitrarily small positive number. We explicitly give the convergence rate constants in terms of the number

[1]Here, connected on average means that the graph that supports the non-zero entries of the expected weight matrix $\mathbb{E}[W(k)]$ is connected.

[2]Note that we assume deterministic cost functions f_i's; the randomness is only due to the underlying random networks.

of nodes N and the network statistics, more precisely, in terms of the quantity $\overline{\mu} := \left(\|\mathbb{E}[W(k)^2] - J\|\right)^{1/2}$ (See ahead paragraph with heading Notation.)

It is interesting to compare the original algorithms D–NG and D–NC in Chapter 2 with their modified variants, mD–NG and mD–NC, respectively. Clearly, algorithms mD–NG and mD–NC apply for static networks as well. A simulation example in Section 3.8 shows that D–NG (from Chapter 2) may diverge in the presence of link failures, while mD–NG may converge at a slightly lower rate than D–NG on static deterministic networks. Also, mD–NG requires an additional (d-dimensional) vector communication per iteration k. Hence, the modified variant mD–NG compromises slightly the speed of convergence (in terms of the overall number of communications) for robustness with respect to (wrt) D-NG.

As for mD–NC and D–NC, mD–NC utilizes one inner consensus algorithm with $2d$-dimensional variables per outer iteration k, while D–NC has two consensus algorithms with d-dimensional variables. Both D–NC variants converge in our simulations (in the presence of link failures), showing very similar performance.

Technically, the analysis of mD–NG and mD–NC methods is very different from D–NG and D–NC in terms of the disagreement estimates. Namely, the time-varying systems that underly the dynamics of disagreements here require a different analysis from the time varying systems in Chapter 2. Chapter 3 establishes novel bounds on certain products of time-varying matrices to analyze these new dynamics. In terms of the optimality gap analysis, Chapter 3 uses similar tools as Chapter 2.

Brief comment on the literature

We comment on the literature relevant to the analysis of convergence rates of distributed gradient algorithms under random networks. References [2, 20, 23] prove convergence of their algorithms under deterministically time varying or random networks. Typically, these references assume f_i's that are convex, (possibly) non-differentiable, and with bounded gradients over the constraint set. For bounded gradients over the constraint set and random networks, reference [12] establishes $O\left(\log k/\sqrt{k}\right)$ convergence rate (with high probability) of a version of the distributed dual averaging method. With respect to these methods, we assume a more restricted class $\mathcal{F}$ of cost functions–f_i's that are convex and have Lipschitz continuous and bounded gradients, but, in contradistinction, we establish strictly faster convergence rates–at least $O(\log k/k)$ that are not achievable by standard distributed gradient methods [2] on the same class $\mathcal{F}$. (See Chapter 2 for details.) Reference [29] analyzes its accelerated distributed proximal gradient method for *deterministically* time varying networks; in contrast, we deal here with *randomly varying networks*. (For a detailed comparison with [29], see Chapter 2.)

Chapter organization. The next paragraph introduces notation. Section 3.2 introduces the random network and optimization models that we assume and presents the mD–NG algorithm and its convergence rate. Section 3.4 proves the convergence rate of mD–NG. Section 3.5 presents the mD–NC algorithm and states its convergence rate, while Section 3.6 proves the convergence rate result. Section 3.7 discusses extensions and corollaries of our results. Section 3.8 illustrates the mD–NG and mD–NC methods on a Huber loss example. We conclude in Section 3.9. Auxiliary proofs are in Appendix B.

Notation. We denote by $\mathbb{R}^d$ the d-dimensional real coordinate space. We index by a subscript i a (possibly) vector quantity assigned to node i; e.g., $x_i(k)$ is node i's estimate at iteration k. Further, we denote by: A_{lm} or $[A]_{lm}$ the entry in the l-th row and m-th column of matrix A; $A^\top$ the transpose of matrix A; $[a]_{l:m}$ the selection of the l-th, $(l+1)$-th, ..., m-th entries of a vector a; I, 0, $\mathbf{1}$, and e_i, respectively, the identity matrix, the zero matrix, the column vector with unit entries, and the i-th column of I; J the $N \times N$ ideal consensus matrix $J := (1/N)\mathbf{1}\mathbf{1}^\top$; $\otimes$ the Kronecker product of matrices; $\|\cdot\|_l$ the vector (respectively, matrix) l-norm of its argument; $\|\cdot\| = \|\cdot\|_2$ the Euclidean (respectively, spectral) norm of its vector (respectively, matrix) argument ($\|\cdot\|$ also denotes the modulus of a scalar); $\lambda_i(\cdot)$ the i-th smallest *in modulus* eigenvalue; $A \succ 0$ a positive definite Hermitian matrix A; $\lfloor a \rfloor$ the integer part of a real scalar a; $\nabla\phi(x)$ and $\nabla^2\phi(x)$ the gradient and Hessian at x of a twice differentiable function $\phi : \mathbb{R}^d \to \mathbb{R}$, $d \geq 1$; $\mathbb{P}(\cdot)$ and $\mathbb{E}[\cdot]$ the probability and expectation, respectively; and $\mathcal{I}_{\mathcal{A}}$ the indicator of event $\mathcal{A}$. For two positive sequences η_n and χ_n, we have: $\eta_n = O(\chi_n)$ if $\limsup_{n\to\infty} \frac{\eta_n}{\chi_n} < \infty$; $\eta_n = \Omega(\chi_n)$ if $\liminf_{n\to\infty} \frac{\eta_n}{\chi_n} > 0$; and $\eta_n = \Theta(\chi_n)$ if $\eta_n = O(\chi_n)$; and $\eta_n = \Omega(\chi_n)$.

The material in this Chapter has been submitted for publication in [47].

3.2 Algorithm mD–NG

Subsection 7.2.1 introduces the network and optimization models, Subsection 3.2.2 presents the mD–NG algorithm, and Subsection 3.5.2 states our result on its convergence rate.

3.2.1 Problem model

Random network model

The network is random, either due to random link failures or due to the random communication protocol used (e.g., gossip, [10, 79].) Formally, the network is defined by a sequence $\{W(k)\}_{k=1}^{\infty}$ of $N \times N$ random weight matrices that obey the following.

Assumption 3.1 (Random network) The random network satisfies:

(a) The sequence $\{W(k)\}_{k=1}^{\infty}$ is i.i.d.

(b) With probability one, the random matrices $W(k)$ are symmetric, stochastic, and have strictly positive diagonal entries.

(c) There exists a positive scalar $\underline{w}$ such that, for all $i, j = 1, ..., N$, with probability one: $W_{ij}(k) \notin (0, \underline{w})$.

(In Assumption 3.1, the underline notation in $\underline{w}$ indicates that $\underline{w}$ is the *lowest value* that the entries of $W(k)$ can take whenever they are positive.) The off-diagonal entries $W_{ij}(k)$, $i \neq j$, may take the value zero. Assumption 3.1 (c) ensures that a node gives a non-negligible weight to itself (By Assumptions 3.1 (b) and (c), $W_{ii}(k) \geq \underline{w}$, with probability one, $\forall i$); also, whenever $W_{ij}(k) > 0$, i.e., nodes i and j communicate, they assign to each other a non-negligible weight (at least $\underline{w}$).

We denote by $\overline{W} := \mathbb{E}[W(k)]$. Further, define the supergraph $\mathcal{G} = (\mathcal{N}, E)$, where $\mathcal{N}$ is the set of N nodes and $E = \big\{\{i, j\} : i < j, \, \overline{W}_{ij} > 0\big\}$. In words, $\mathcal{G}$ collects all the pairs $\{i, j\}$ for which $W_{ij}(k)$ is nonzero with a positive probability – all realizable communication links.

An example of the model of the $W(k)$'s subsumed by Assumption 3.1 is the link failure model. Here, each link $\{i, j\} \in E$ at time k is a Bernoulli random variable; when it takes the value one, the link $\{i, j\}$ is interpreted as being online (communication occurs), and, when it equals zero, the link fails (communication does not occur). The links (the corresponding Bernoulli variables) are independent over time, but may be correlated in space. A possible weight assignment is to set: 1) for $i \neq j$, $\{i, j\} \in E$: $W_{ij}(k) = w_{ij} = 1/N$, when $\{i, j\}$ is online, and $W_{ij}(k) = 0$, else; 2) for $i \neq j$, $\{i, j\} \notin E$: $W_{ij}(k) \equiv 0$; and 3) $W_{ii}(k) = 1 - \sum_{j \neq i} W_{ij}(k)$. An alternative assignment, when the link occurrence probabilities and their correlations are known, is to set the (possibly mutually different) weights w_{ij}, $\{i, j\} \in E$, as the minimizers of $\overline{\mu}$ (See Section 3.8 and Chapter 5.)

We further make the following Assumption.

Assumption 3.2 (Network connectedness) The supergraph $\mathcal{G}$ is connected.

Denote by $\widetilde{W}(k) = W(k) - J$, where we recall $J := (1/N)\mathbf{1}\mathbf{1}^{\top}$, by

$$\widetilde{\Phi}(k, t) = \widetilde{W}(k)...\widetilde{W}(t+2), \;\; t = 0, 1, ..., k-2, \tag{3.1}$$

and by $\widetilde{\Phi}(k, k-1) = I$.

Recall $\overline{\mu} := \left(\|\mathbb{E}\left[W^2(k)\right] - J\|\right)^{1/2}$. One can show that $\overline{\mu}$ equals the square root of the second largest eigenvalue of $\mathbb{E}\left[W^2(k)\right]$ and that, under Assumptions 3.1 and 3.2, $\overline{\mu} < 1$. Lemma 3.3 below demonstrates that $\overline{\mu}$ characterizes the geometric decay (in $k-t$) of the first and second moments of $\widetilde{\Phi}(k,t)$. The proof of Lemma 3.3 is in the Appendix.

Lemma 3.3 Let Assumptions 3.1 and 3.2 hold. Then:

$$\mathbb{E}\left[\left\|\widetilde{\Phi}(k,t)\right\|\right] \leq N\,\overline{\mu}^{k-t-1} \tag{3.2}$$

$$\mathbb{E}\left[\left\|\widetilde{\Phi}(k,t)^\top\widetilde{\Phi}(k,t)\right\|\right] \leq N^2\,\overline{\mu}^{2(k-t-1)} \tag{3.3}$$

$$\mathbb{E}\left[\left\|\widetilde{\Phi}(k,s)^\top\widetilde{\Phi}(k,t)\right\|\right] \leq N^3\,\overline{\mu}^{(k-t-1)+(k-s-1)}, \tag{3.4}$$

for all $t, s = 0, ..., k-1$, for all $k = 1, 2, ...$

The bounds in (3.2)-(3.4) may be loose, and could be easily improved for certain values of the arguments s and t, but as stated they are enough to prove the results below, while simplifying the presentation.

For static networks, $W(k) \equiv W$, where W is a doubly stochastic, deterministic, symmetric weight matrix W. In this case, the quantity $\overline{\mu} := \|W - J\|$ equals the spectral gap, i.e., the modulus of the second largest (in modulus) eigenvalue of W. Further, for static networks, the constants N, N^2, and N^3 in (3.2)–(3.4) can be reduced to unity.

Finally, we remark that the requirement of the entries of $W(k)$ being nonnegative with probability one can be relaxed, as long as $\overline{\mu}$ is less than one. We refer to Chapter 5 for details.

Optimization model

We now introduce the optimization model. The nodes solve the unconstrained problem:

$$\text{minimize} \quad \sum_{i=1}^{N} f_i(x) =: f(x). \tag{3.5}$$

The function $f_i : \mathbb{R}^d \to \mathbb{R}$ is known only to node i. We impose the following three Assumptions.

Assumption 3.4 (Solvability) There exists a solution $x^\star \in \mathbb{R}^d$ such that $f(x^\star) = f^\star := \inf_{x\in\mathbb{R}^d} f(x)$.

Assumption 3.5 (Lipschitz continuous gradient) For all i, f_i is convex and has Lipschitz continuous gradient

with constant $L \in [0, \infty)$:

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|, \quad \forall x, y \in \mathbb{R}^d.$$

Assumption 3.6 (Bounded gradients) There exists a constant $G \in [0, \infty)$ such that, $\forall i$, $\|\nabla f_i(x)\| \leq G$, $\forall x \in \mathbb{R}^d$.

We comment on the optimization model in Assumptions 3.4–3.6. Assumptions 3.4 and 3.5 are standard in the analysis of gradient methods; in particular, Assumption 3.5 is precisely the Assumption required by the centralized Nesterov gradient method [11]. Assumption 3.6 is in addition to what is common in the centralized Nesterov gradient method. Chapter 2 demonstrates that (even on) static networks and a constant $W(k) \equiv W$, the convergence rate of D–NG becomes arbitrarily slow if Assumption 3.6 is violated (see Chapter 2 for a precise statement.) We remark that this requirement is not specific to our accelerated methods but is also a feature of, e.g., the standard distributed gradient method in [2] (see Chapter 2.)

3.2.2 Algorithm mD–NG for random networks

We modify the D–NG algorithm in Chapter 2 to handle random networks. Each node i maintains its solution estimate $x_i(k)$ and an auxiliary variable $y_i(k)$ over iterations $k = 0, 1, ...$ Node i uses arbitrary initialization $x_i(0) = y_i(0) \in \mathbb{R}^d$ and performs the following updates for $k = 1, 2, ...$

$$x_i(k) = \sum_{j \in O_i(k)} W_{ij}(k)\, y_j(k-1) - \alpha_{k-1} \nabla f_i(y_i(k-1)) \tag{3.6}$$

$$y_i(k) = (1 + \beta_{k-1})\, x_i(k) - \beta_{k-1} \sum_{j \in O_i(k)} W_{ij}(k)\, x_j(k-1). \tag{3.7}$$

In (3.6)–(3.7), $O_i(k) = \{j \in \{1, ..., N\} : W_{ij}(k) > 0\}$ is the (random) neighborhood of node i (including node i) at time k. For $k = 0, 1, 2, ...$, the step-size α_k is:

$$\alpha_k = c/(k+1), \;\; c \leq 1/(2L). \tag{3.8}$$

We adopt the sequence β_k from the centralized Nesterov gradient method [11]:

$$\beta_k = \frac{k}{k+3}. \tag{3.9}$$

The mD–NG algorithm works as follows. At iteration k, node i receives the variables $x_j(k-1)$ and

$y_j(k-1)$ from its current neighbors $j \in O_i(k) - \{i\}$, and updates $x_i(k)$ and $y_i(k)$ via (3.6) and (3.7). We assume that all nodes know the constant L (or its upper bound) beforehand to set α_k in (3.8). We show in Section 3.7 how this requirement can be relaxed.

The mD–NG algorithm given by (3.6) and (3.7) differs from the original D–NG in Chapter 2 in step (3.7). With D–NG, nodes communicate only the variables $y_j(k-1)$'s; with mD–NG, they also communicate the $x_j(k-1)$'s. As we will show, the latter modification allows for the robustness to link failures. (See also Theorems 3.7 and 3.8 and the simulations in Section 3.8.) Further, the mD–NG does not require that the weight matrix be positive definite, as required by D–NG in Chapter 2.

Vector form. We rewrite mD–NG in vector form. Introduce $x(k) := (x_1(k)^\top, ..., x_N(k)^\top)^\top$, $y(k) := (y_1(k)^\top, ..., y_N(k)^\top)^\top$, and $F : \mathbb{R}^{Nd} \to \mathbb{R}$, $F(x_1, ..., x_N) := f_1(x_1) + ... + f_N(x_N)$. Then, mD–NG algorithm in vector form is:

$$x(k) = (W(k) \otimes I)\ y(k-1) - \alpha_{k-1} \nabla F(y(k-1)) \tag{3.10}$$

$$y(k) = (1 + \beta_{k-1})\, x(k) - \beta_{k-1}\ (W(k) \otimes I)\ x(k-1), \tag{3.11}$$

$k = 1, 2, ...$, with $x(0) = y(0) \in \mathbb{R}^{Nd}$, where $W(k) \otimes I$ is the Kronecker product of $W(k)$ and the $d \times d$ identity matrix.

Initialization. For simplicity of notation, and without loss of generality (wlog), we assume throughout, with all proposed methods, that nodes initialize their estimates to the same values, i.e., $x_i(0) = y_i(0) = x_j(0) = y_j(0)$, for all i, j; for example, $x_i(0) = y_i(0) = x_j(0) = y_j(0) = 0$.

3.2.3 Convergence rate of mD–NG

In this Subsection, we state our convergence rate result for mD–NG distributed method and random networks. Proofs are in Section 3.4.

We estimate the expected (normalized) optimality gap in the cost[3] at each node i: $\frac{1}{N}\mathbb{E}\left[f(x_i) - f^\star\right] = \frac{1}{N}\left(\mathbb{E}\left[f(x_i)\right] - f^\star\right)$, where x_i is node i's solution estimate at a certain stage of the algorithm. We are interested in how the node i's optimality gap depends (decreases) with: 1) the number of per-node gradient evaluations k (iterations); and 2) the total number of $2\,d$-dimensional vector communications per node $\mathcal{K}$. With mD–NG, we have that $k = \mathcal{K}$ – at each iteration k, there is one and only one per-node $2\,d$-dimensional communication and one per-node gradient evaluation. With mD–NC, as we will see, there are several per-node $2\,d$-dimensional communications at each k.

[3] We normalize the optimality gap by N, as is frequently done in the literature, e.g., [12, 27].

With both methods, we also establish the mean square convergence rate on the disagreements of different node estimates, in terms of k and $\mathcal{K}$, showing that the mean square disagreement converges to zero.

Denote by $\overline{x}(k) := \frac{1}{N}\sum_{i=1}^{N} x_i(k)$ and $\overline{y}(k) := \frac{1}{N}\sum_{i=1}^{N} y_i(k)$ the network-wide global averages of the nodes' estimates. Introduce the disagreements: $\widetilde{x}_i(k) = x_i(k) - \overline{x}(k)$ and $\widetilde{x}(k) = \left(\widetilde{x}_1(k)^\top, ..., \widetilde{x}_N(k)^\top\right)^\top$, and the analogous quantities $\widetilde{y}_i(k)$ and $\widetilde{y}(k)$. Denote by $\widetilde{z}(k) := \left(\widetilde{y}(k)^\top,\, \widetilde{x}(k)^\top\right)^\top$. We have the following Theorem on $\mathbb{E}\left[\|\widetilde{z}(k)\|\right]$ and $\mathbb{E}\left[\|\widetilde{z}(k)\|^2\right]$. Note that $\|\widetilde{x}(k)\| \leq \|\widetilde{z}(k)\|$, and so $\mathbb{E}\left[\|\widetilde{x}(k)\|\right] \leq \mathbb{E}\left[\|\widetilde{z}(k)\|\right]$ and $\mathbb{E}\left[\|\widetilde{x}(k)\|^2\right] \leq \mathbb{E}\left[\|\widetilde{z}(k)\|^2\right]$. (The same inequalities hold for $\widetilde{y}(k)$ as well.) Recall also $\overline{\mu}$ in Lemma 3.3. Theorem 3.7 states that the mean square disagreement of different nodes' estimates converges to zero at rate $1/k^2$.

Theorem 3.7 Consider the mD–NG algorithm given in (3.6)–(3.9) under Assumptions 3.1–3.6. Then, for all $k = 1, 2, ...$

$$\mathbb{E}\left[\|\widetilde{z}(k)\|\right] \;\leq\; \frac{50\, c\, N^{3/2}\, G}{(1-\overline{\mu})^2}\,\frac{1}{k} \tag{3.12}$$

$$\mathbb{E}\left[\|\widetilde{z}(k)\|^2\right] \;\leq\; \frac{50^2\, c^2\, N^4 G^2}{(1-\overline{\mu})^4\, k^2}. \tag{3.13}$$

Theorem 3.8 establishes the $O(\log k/k)$ (and $O(\log \mathcal{K}/\mathcal{K})$) convergence rate of mD–NG.

Theorem 3.8 Consider the mD–NG algorithm in (3.6)–(3.9) under Assumptions 3.1–3.6. Let $\|\overline{x}(0) - x^\star\| \leq R$, $R \geq 0$. Then, at any node i, the expected normalized optimality gap $\frac{1}{N}\mathbb{E}\left[f(x_i(k)) - f^\star\right]$ is $O(\log k/k)$; more precisely:

$$\frac{\mathbb{E}\left[f(x_i(k)) - f^\star\right]}{N} \leq \frac{2\,R^2}{c}\,\frac{1}{k} + \frac{50^2\, c^2\, N^3\, L\, G^2}{(1-\overline{\mu})^4}\,\frac{1}{k}\sum_{t=1}^{k-1}\frac{(t+2)^2}{(t+1)t^2} + \frac{50\, N^2\, c\, G^2}{(1-\overline{\mu})^2}\,\frac{1}{k}. \tag{3.14}$$

Remark. For the optimized step-size, $c = \Theta\left(\frac{(1-\overline{\mu})^{4/3}}{N}\right)$, we obtain that

$$\frac{\mathbb{E}\left[f(x_i(k)) - f^\star\right]}{N} = O\left(\frac{N}{(1-\overline{\mu})^{4/3}}\frac{\log k}{k}\right),\;\; \forall i.$$

Remark. For static networks, when $W(k) \equiv W$ and $\overline{\mu} := \|W - J\|$, the factors N and N^3 in (3.14) reduce to unity. Theorem 3.8 is similar to Theorem 5 (a) in Chapter 2. Comparing the two Theorems (for static networks), we can see that the rate constant in Theorem 3.8 above depends on $\overline{\mu}$ as $O\left(\frac{1}{(1-\overline{\mu})^4}\right)$; with Theorem 2.8 (a) in Chapter 2, this dependence is $O(\frac{1}{(1-\overline{\mu})^{3+\xi}})$, for arbitrarily small positive quantity ξ. Hence, with respect to D–NG, mD–NG method has an increased (worse) theoretical constant, but exhibits

robustness to link failures. (See Section 3.8.)

3.3 Intermediate results

This section establishes intermediate results on certain scalar sums and the products of time-varying 2×2 matrices that arise in the analysis of the mD–NG and mD–NC methods.

Scalar sums

We have the following Lemma.

Lemma 3.9 Consider a nonnegative scalar $r < 1$. Then, for all $k = 1, 2, ...$:

$$\sum_{t=1}^{k} r^t \, t \;\; \leq \;\; \frac{r}{(1-r)^2} \leq \frac{1}{(1-r)^2} \tag{3.15}$$

$$\sum_{t=0}^{k-1} r^{k-t-1} \frac{1}{t+1} \;\; \leq \;\; \frac{1}{(1-r)^2 \, k}. \tag{3.16}$$

Proof:

We first show (3.15). Denote by $\frac{d}{dr} h(r)$ the first derivative of a function $h(r)$, $h : \mathbb{R} \to \mathbb{R}$. We have:

$$\begin{aligned} \sum_{t=1}^{k} r^t \, t &= r \sum_{t=1}^{k} r^{t-1} t = r \, \frac{d}{dr} \left(\sum_{t=1}^{k} r^t \right) \\ &= r \, \frac{d}{dr} \left(\frac{r - r^{k+1}}{1-r} \right) = \frac{r \left(1 - (k+1) r^k (1-r) - r^{k+1} \right)}{(1-r)^2} \\ &\leq \frac{r}{(1-r)^2}, \;\; \forall k = 1, 2, ... \end{aligned}$$

Thus, the bound in (3.15).

To obtain the second inequality (3.16), we use $k/(t+1) \leq k - t$, $\forall t = 0, 1, ..., k-1$. Using the latter and (3.15):

$$\sum_{t=0}^{k-1} r^{k-t-1} \frac{1}{t+1} = \frac{1}{k} \sum_{t=0}^{k-1} r^{k-t-1} \frac{k}{t+1} \leq \frac{1}{k \, r} \sum_{t=0}^{k-1} r^{k-t} (k-t) \leq \frac{1}{k} \, \frac{1}{(1-r)^2}. \;\; \square$$

Products of matrices $B(k)$

Consider the matrix $B(k)$, defined, for $k = 1, 2, ...$, by:

$$B(k) := \begin{bmatrix} (1+\beta_{k-1}) & -\beta_{k-1} \\ 1 & 0 \end{bmatrix}, \tag{3.17}$$

where β_{k-1} is in (3.9). As we will see, the proofs of Theorems 3.7 and 3.14 rely much on the analysis of products $\mathcal{B}(k,t)$, defined by:

$$\mathcal{B}(k,t) := B(k)B(k-1)...B(k-t),\ t = 0, 1, ..., k-2, \tag{3.18}$$

and $\mathcal{B}(k,-1) := I$.

We first give a Lemma that explicitly calculates the product $\mathcal{B}(k,t)$, $t = 1, 2, ..., k-2$, for $k \geq 3$.

Lemma 3.10 Let $k \geq 3$, consider $\mathcal{B}(k,t)$ in (3.18), define $a_t := 3/(t+3)$, $t = 0, 1, ...$, and let:

$$B_1 := \begin{bmatrix} 2 & -1 \\ 1 & 0 \end{bmatrix},\ B_2 := \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix},\ B_3 := \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix}. \tag{3.19}$$

Then, for $t = 1, 2, ..., k-2$:

$$\mathcal{B}(k,t) = B_1^{t+1} - \sigma_2(k,t)\, B_2 - \sigma_3(k,t)\, B_3,$$

where

$$\sigma_2(k,t) = a_{k-t-1}\, t + a_{k-t}\beta_{k-t-1}\,(t-1) + a_{k-t+1}\beta_{k-t}\beta_{k-t-1}(t-2) + ... + a_{k-2}\beta_{k-3}...\beta_{k-t-1} \tag{3.20}$$

$$\sigma_3(k,t) = a_{k-t-1} + a_{k-t}\beta_{k-t-1} + a_{k-t+1}\beta_{k-t}\beta_{k-t-1} + ... + a_{k-2}\beta_{k-3}...\beta_{k-t-1}. \tag{3.21}$$

Proof: The proof is by mathematical induction on $t = 1, 2, ..., k-2$. First, we verify that the claim

holds for $t = 1$. We have:

$$\begin{aligned}
\mathcal{B}(k,1) &:= B(k)B(k-1) = (B_1 - a_{k-1}B_3)(B_1 - a_{k-2}B_3) \\
&= B_1^2 - a_{k-1}\, B_3B_1 - a_{k-2}B_1B_3 + a_{k-1}a_{k-2}B_3^2 \\
&= B_1^2 - a_{k-2}\, B_2 - (a_{k-1} + a_{k-2} - a_{k-1}a_{k-2})B_3 \\
&= B_1^2 - a_{k-2}B_2 - (a_{k-2} + a_{k-1}\beta_{k-2})B_3 \\
&= B_1^2 - \sigma_2(k,1)\, B_2 - \sigma_3(k,1)\, B_3,
\end{aligned}$$

where the second equality uses $B_3B_1 = B_3$, $B_1B_3 = B_2 + B_3$, and $B_3^2 = B_3$, and the fourth equality uses $1 - a_{k-2} = \beta_{k-2}$. Thus, the claim for $t = 1$ holds. Now, suppose the claim holds true for some fixed t, $t \in \{1, ..., k-3\}$. We must show that it holds true for $t+1$ as well. Using the inductive hypothesis and the definition of $\mathcal{B}(k, t+1)$:

$$\begin{aligned}
\mathcal{B}(k,t+1) &= \mathcal{B}(k,t)\, B(k-t-1) \\
&= (B_1^{t+1} - \sigma_2(k,t)B_2 - \sigma_3(k,t)B_3)\, (B_1 - a_{k-t-2}B_3) \\
&= B_1^{t+2} - \sigma_2(k,t)B_2 - \sigma_3(k,t)B_3 \qquad (3.22) \\
&\qquad -a_{k-t-2}((t+1)B_2 + B_3) + \sigma_2(k,t)a_{k-t-2}B_2 + \sigma_3(k,t)a_{k-t-2}B_3 \\
&= B_1^{t+2} - (\sigma_2(k,t)\beta_{k-t-2} + (t+1)a_{k-t-2})B_2 - (\sigma_3(k,t)\beta_{k-t-2} + a_{k-t-2})B_3.
\end{aligned}$$

Equality (3.22) uses $B_2B_1 = B_2$, $B_3^2 = B_3$, $B_2B_3 = B_2$, and the fact that $B_1^{t+1}B_3 = (t+1)B_2 + B_3$. (This is trivial to show by mathematical induction on t.) Next, recognize from (3.20)–(3.21) that $\sigma_2(k, t+1) = \sigma_2(k,t)\beta_{k-t-2} + (t+1)a_{k-t-2}$, and $\sigma_3(k, t+1) = \sigma_3(k,t)\beta_{k-t-2} + a_{k-t-2}$. Thus, the result. □

Next, we establish the bounds on the sums $\sigma_2(k,t)$ and $\sigma_3(k,t)$.

Lemma 3.11 Consider $\sigma_2(k,t)$ and $\sigma_3(k,t)$ in (3.20) for $t = 1, ..., k-2$, $k \geq 3$. Then:

$$\frac{t^2}{k+2} \leq \sigma_2(k,t) \leq t+1, \qquad 0 \leq \sigma_3(k,t) \leq 1. \qquad (3.23)$$

Proof:

We prove each of the four inequalities above.

Proof of the right inequality on $\sigma_2(k,t)$

We conduct induction on $t = 1, ..., k-2$. For $t = 1$, we have that $\sigma_2(k,1) = a_{k-2} = 3/(k+1) \leq 1+1$, $\forall k$, and so the claim holds for $t = 1$. Suppose that the claim is true for some fixed $t \geq 1$. Further, note that $\sigma_2(k,t)$ can be written recursively as:

$$\sigma_2(k,t+1) = a_{k-t-2}(t+1) + \beta_{k-t-2}\sigma_2(k,t),\ t = 1, ..., k-3. \tag{3.24}$$

Using (3.24) and the induction hypothesis: $\sigma_2(k,t+1) \leq (t+1)a_{k-t-2} + \beta_{k-t-2}(t+1) = (a_{k-t-2} + \beta_{k-t-2})(t+1) = t+1 \leq t+2$. Thus, the right inequality on $\sigma_2(k,t)$.

Proof of the left inequality on $\sigma_2(k,t)$

We perform the proof by mathematical induction on t. First, we verify the claim for $t = 1$:

$$\sigma_2(k,1) = a_{k-2} = \frac{3}{k+1} \geq \frac{1^2}{k+2},$$

and so the claim for $t = 1$ holds. Now, suppose that the claim is true for some $t \in \{1, 2, ..., k-3\}$, i.e.:

$$\sigma_2(k,t) \geq \frac{t^2}{k+2}. \tag{3.25}$$

We must show that $\sigma_2(k,t+1) \geq \frac{(t+1)^2}{k+2}$. Using (3.24):

$$\begin{aligned} \sigma_2(k,t+1) &\geq a_{k-t-2}(t+1) + \beta_{k-t-2}\frac{t^2}{k+2} \\ &= \frac{(t+1)^2}{k+2} + \frac{t(k-t) + (2k+5t+5)}{(k+2)(k-t+1)} \geq \frac{(t+1)^2}{k+2}, \end{aligned}$$

where the last equality follows after some algebraic manipulations. By induction, the last inequality completes the proof of the lower bound on $\sigma_2(k,t)$.

Proof of the lower bound on $\sigma_3(k,t)$. is trivial.

Proof of the upper bound on $\sigma_3(k,t)$

We again proceed by induction. For $t = 1$, we have:

$$\sigma_3(k,1) = a_{k-2} + a_{k-1}\beta_{k-2} \leq a_{k-2} + \beta_{k-2} = 1.$$

Suppose now that the claim holds true for some $t \in \{1, ..., k-3\}$, i.e.:

$$\sigma_3(k,t) \leq 1.$$

Note from (3.20) that:

$$\sigma_3(k,t+1) = \beta_{k-t-2}\,\sigma_3(k,t) + a_{k-t-2}.$$

Thus, using the induction hypothesis:

$$\sigma_3(k,t+1) \leq \beta_{k-t-2} + a_{k-t-2} \leq 1,$$

and the proof of the upper bound on $\sigma_3(k,t)$ is completed. □

We are now ready to upper bound $\|\mathcal{B}(k, k-t-2)\|$, the result of direct use in proving Theorem 3.7.

Lemma 3.12 Consider the product $\mathcal{B}(k,t)$ defined in (3.18). Then, for all $t = 0, ..., k-1$, for all $k = 1, 2, ...$

$$\|\mathcal{B}(k, k-t-2)\| \leq 8\frac{(k-t-1)(t+1)}{k} + 5. \tag{3.26}$$

Proof:

Fix some $t \in \{1, ..., k-2\}$, $k \geq 3$, and consider the explicit expression for $\mathcal{B}(k,t)$ in Lemma 3.10. It is easy to show that $B_1^t = t\,B_2 + I$. Thus,

$$\mathcal{B}(k,t) = (t+1-\sigma_2(k,t))\;B_2 + I - \sigma_3(k,t)B_3. \tag{3.27}$$

By Lemma 3.11, the term:

$$0 \leq t+1-\sigma_2(k,t) \leq t+1-t^2/(k+2).$$

Next, use the latter equation; $\sigma_3(k,t) \leq 1$ (by Lemma 3.11); and $\|B_2\| = 2$, $\|B_3\| = \sqrt{2} < 2$. Applying these findings to (3.27), we obtain:

$$\|\mathcal{B}(k,t)\| \leq 2\left(t+1-\frac{t^2}{k+2}\right) + 3 = 2\left(t-\frac{t^2}{k+2}\right) + 5, \tag{3.28}$$

for all $t = 1, 2, ..., k-2$, $k \geq 3$. Next, we set the second argument of $\mathcal{B}(k,\cdot)$ to $k-t-2$, $t = 0, ..., k-3$,

$k \geq 3$. From (3.28), we obtain:

$$\begin{aligned} \|\mathcal{B}(k,k-t-2)\| &\leq 2\left(k-t-2-\frac{(k-t-2)^2}{k+2}\right)+5 \\ &= 2(k-t-2)\frac{t+4}{k+2}+5 \leq 8(k-t-1)\frac{t+1}{k}+5, \end{aligned}$$

for all $t = 0, ..., k-3$, for all $k \geq 3$. (Here, we used $(t+4)/(k+2) \leq 4(t+1)/k$.) Thus, we have obtained the desired inequality (3.26) for $t = 0, ..., k-3$, for $k \geq 3$. To complete the proof, we show that (3.26) holds also for: 1) $t = k-2$, $k \geq 2$; 2) $t = k-1$, $k \geq 1$. Consider first case 1 and $\mathcal{B}(k, k-2) = B(k-1) = B_1 - a_{k-1}B_3$, $k \geq 2$. We have $\|\mathcal{B}(k,k-2)\| \leq \|B_1\| + \|B_3\| < 5$, and so (3.26) holds for $t = k-2$, $k \geq 2$. Next, consider case 2 and $\mathcal{B}(k,k-1) = I$, $k \geq 1$. We have that $\|\mathcal{B}(k,k-1)\| = 1 < 5$, and so (3.26) also holds for $t = k-1$, $k \geq 1$. This completes the proof of the Lemma. □

3.4 Proofs of Theorems 3.7 and 3.8

Subsection 3.4.1 proves Theorem 3.7, while Subsection 3.4.2 proves Theorem 3.8.

3.4.1 Proof of Theorem 3.7

We proceed by proving Theorem 3.7. Throughout this proof and the rest of the chapter, we establish certain equalities and inequalities on random quantities of interest. All of these equalities and inequalities further ahead hold either: 1) surely, for any random realization, or: 2) in expectation (From the notation, it is clear which of the two cases is in force.) For notational simplicity, we perform the proof of Theorem 3.7 for the case $d = 1$, but the proof extends for a generic $d > 1$. The proof has three steps. In Step 1, we derive the dynamic equation for the disagreement $\widetilde{z}(k) = \left(\widetilde{y}(k)^\top, \widetilde{x}(k)^\top\right)^\top$. In Step 2, we unwind the dynamic equation, expressing $\widetilde{z}(k)$ in terms of the products $\widetilde{\Phi}(k,t)$ in (3.1) and $\mathcal{B}(k,t)$ in (3.18). Finally, in Step 3, we apply the already established bounds on the norms of the latter products.

Step 1. Disagreement dynamics

Denote by $\widetilde{z}(k) := \left(\widetilde{y}(k)^\top, \widetilde{x}(k)^\top\right)^\top$. Multiplying (3.6)–(3.7) from the left by $(I-J)$, and using $(I-J)W(k) = \widetilde{W}(k) - J$, obtain:

$$\widetilde{z}(k) = \left(B(k) \otimes \widetilde{W}(k)\right) \widetilde{z}(k-1) + u(k-1), \;\; k = 1, 2, ..., \tag{3.29}$$

and $\widetilde{z}(0) = 0$, where

$$u(k-1) = -\begin{bmatrix} \alpha_{k-1}(1+\beta_{k-1})(I-J)\nabla F(y(k-1)) \\ \alpha_{k-1}(I-J)\nabla F(y(k-1)) \end{bmatrix}. \tag{3.30}$$

Step 2. Unwinding recursion (3.29)

Recall $\widetilde{\Phi}(k,t)$ in (3.1), and $\mathcal{B}(k,t)$ in (3.18). Then, unwinding (3.29), and using the Kronecker product property $(A \otimes B)(C \otimes D) = (AB) \otimes (CD)$, we obtain:

$$\widetilde{z}(k) = \sum_{t=0}^{k-1} \Big(\mathcal{B}(k, k-t-2) \otimes \widetilde{\Phi}(k,t) \Big)\, u(t), \tag{3.31}$$

for all $k = 1, 2, ...$ Note that the quantities $u(t)$ and $\widetilde{\Phi}(k,t)$ in (3.31) are random, while the $\mathcal{B}(k, k-t-2)$'s are deterministic.

Step 3. Finalizing the proof

Consider $u(t)$ in (3.30). By Assumption 3.6, we have $\|\nabla F(y(t))\| \leq \sqrt{N}G$. Using the latter, the step-size $\alpha_t = c/(t+1)$, and $\|I - J\| = 1$, we obtain that $\|u(t)\| \leq \frac{\sqrt{3}\, c \sqrt{N}\, G}{t+1}$, for any random realization of $u(t)$. Use the latter bound, Lemma 3.12, and the sub-multiplicative and sub-additive properties of norms; from (3.31) we obtain:

$$\|\widetilde{z}(k)\| \leq \Big(8\sqrt{3}\, c\, \sqrt{N}\, G\Big)\, \frac{1}{k} \sum_{t=0}^{k-1} \|\widetilde{\Phi}(k,t)\|\, (k-t-1) + \Big(5\sqrt{3}\, c\, \sqrt{N}\, G\Big) \sum_{t=0}^{k-1} \|\widetilde{\Phi}(k,t)\|\, \frac{1}{t+1}.$$

Taking expectation, and using Lemma 3.3:

$$\mathbb{E}\left[\|\widetilde{z}(k)\|\right] \leq \Big(8\sqrt{3}\, c\, N^{3/2}\, G\Big)\, \frac{1}{k} \sum_{t=0}^{k-1} \overline{\mu}^{k-t-1}\, (k-t-1) + \Big(5\sqrt{3}\, c\, N^{3/2}\, G\Big) \sum_{t=0}^{k-1} \overline{\mu}^{k-t-1}\, \frac{1}{t+1}.$$

Finally, applying Lemma 3.9 to the last equation with $r = \overline{\mu}$, the result in (3.12) follows.

We now prove (3.13). Consider $\|\widetilde{z}(k)\|^2$. From (3.31), we obtain:

$$\begin{aligned}\|\widetilde{z}(k)\|^2 &= \sum_{t=0}^{k-1}\sum_{s=0}^{k-1} u(t)^\top \left(\mathcal{B}(k,k-t-2)^\top \otimes \widetilde{\Phi}(k,t)^\top\right)\left(\mathcal{B}(k,k-t-2)\otimes\widetilde{\Phi}(k,s)\right) u(s)\\ &= \sum_{t=0}^{k-1}\sum_{s=0}^{k-1} u(t)^\top \left(\mathcal{B}(k,k-t-2)^\top \mathcal{B}(k,k-s-2)\right) \otimes \left(\widetilde{\Phi}(k,t)^\top \widetilde{\Phi}(k,s)\right) u(s),\end{aligned}$$

where the last equality uses the property $(A\otimes B)(C\otimes D) = (AC)\otimes(BD)$. Further, using the sub-additive and sub-multiplicative properties of norms, we obtain:

$$\begin{aligned}\|\widetilde{z}(k)\|^2 &\leq \sum_{t=0}^{k-1}\sum_{s=0}^{k-1} \|\mathcal{B}(k,k-t-2)\|\,\|\mathcal{B}(k,k-s-2)\|\left\|\widetilde{\Phi}(k,t)^\top\widetilde{\Phi}(k,s)\right\|\,\|u(t)\|\,\|u(s)\| \qquad (3.32)\\ &\leq \sum_{t=0}^{k-1}\sum_{s=0}^{k-1}\left(\frac{8(k-t-1)(t+1)}{k}+5\right)\left(\frac{8(k-s-1)(s+1)}{k}+5\right)\left\|\widetilde{\Phi}(k,t)^\top\widetilde{\Phi}(k,s)\right\|\frac{3\,c^2\,NG^2}{(t+1)(s+1)},\end{aligned}$$

where the last inequality uses Lemma 3.12 and $\|u(t)\| \leq \left(\sqrt{3}c\sqrt{N}G\right)/(t+1)$. Taking expectation and applying Lemma 3.3, we obtain:

$$\begin{aligned}\mathbb{E}\left[\|\widetilde{z}(k)\|^2\right] &\leq \left(3\,c^2\,N^4G^2\right)\sum_{t=0}^{k-1}\sum_{s=0}^{k-1}\left(\frac{8(k-t-1)(t+1)}{k}+5\right)\left(\frac{8(k-s-1)(s+1)}{k}+5\right)\\ &\times\ \overline{\mu}^{k-t-1+k-s-1}\,\frac{1}{(t+1)(s+1)}\\ &= \left(3\,c^2\,N^4G^2\right)\left(\sum_{t=0}^{k-1}\left(\frac{8(k-t-1)(t+1)}{k}+5\right)\frac{\overline{\mu}^{k-t-1}}{t+1}\right)^2\\ &\leq \frac{50^2\,c^2\,N^4G^2}{(1-\overline{\mu})^4\,k^2},\end{aligned}$$

where the last inequality applies Lemma 3.9. Thus, the bound in (3.13). The proof of Theorem 3.7 is complete.

3.4.2 Proof of Theorem 3.8

The proof is similar to the proof of Theorem 5 (a) in Chapter 2. Henceforth, we give only a proof outline and refer to Chapter 2 for details. The proof is based on the evolution of the global averages $\overline{x}(k) =$

$\frac{1}{N}\sum_{i=1}^{N} x_i(k)$, and $\overline{y}(k) = \frac{1}{N}\sum_{i=1}^{N} y_i(k)$. Denote by:

$$\begin{aligned}
\widehat{f}_{k-1} &:= \sum_{i=1}^{N}\Big(f_i(y_i(k-1)) + \nabla f_i(y_i(k-1))^\top(\overline{y}(k-1) - y_i(k-1))\Big) \qquad (3.33)\\
\widehat{g}_{k-1} &:= \sum_{i=1}^{N}\nabla f_i\left(y_i(k-1)\right)\\
L_{k-1} &:= \frac{N}{\alpha_{k-1}} = \frac{Nk}{c} \geq 2\,N\,L\,k\\
\delta_{k-1} &:= L\,\|\widetilde{y}(k-1)\|^2.
\end{aligned}$$

Then, it is easy to show that $\overline{x}(k)$, $\overline{y}(k)$ evolve as:

$$\begin{aligned}
\overline{x}(k) &= \overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} \qquad (3.34)\\
\overline{y}(k) &= (1+\beta_{k-1})\,\overline{x}(k) - \beta_{k-1}\overline{x}(k-1), \qquad (3.35)
\end{aligned}$$

$k = 1, 2, ...$, with $\overline{x}(0) = \overline{y}(0)$. As shown in Chapter 2, $\left(\widehat{f}_{k-1}, \widehat{g}_{k-1}\right)$ is a (L_{k-1}, δ_{k-1}) inexact oracle, i.e., it holds:

$$f(x) + \widehat{g}_{k-1}^\top(x - \overline{y}(k-1)) \leq f(x) \leq \widehat{f}_{k-1} + \widehat{g}_{k-1}^\top\left(x - \overline{y}(k-1)\right) + \frac{L_{k-1}}{2}\|x - \overline{y}(k-1)\|^2 + \delta_{k-1}, \quad (3.36)$$

for all points $x \in \mathbb{R}^d$. [4]

Thus, we can apply Lemma 2.5 in Chapter 2, which, using the expression for δ_{k-1} in (3.33), gives:

$$\begin{aligned}
(k+1)^2 &\quad (f(\overline{x}(k)) - f^\star) + \frac{2Nk}{c}\|\overline{v}(k) - x^\star\|^2 \qquad (3.37)\\
&\leq \qquad (k^2-1)\,(f(\overline{x}(k-1)) - f^\star) + \frac{2Nk}{c}\|\overline{v}(k-1) - x^\star\|^2 + (k+1)^2 L\|\widetilde{y}(k-1)\|^2,
\end{aligned}$$

where $\overline{v}(k) = \left(\overline{y}(k) - (1-\theta_k)\overline{x}(k)\right)/\theta_k$. Further, dividing (3.37) by k and unwinding the resulting inequality, it can be shown that:

$$\frac{1}{N}\left(f(\overline{x}(k)) - f^\star\right) \leq \frac{2}{k\,c}\|\overline{x}(0) - x^\star\|^2 + \frac{L}{N\,k}\sum_{t=1}^{k}\frac{(t+1)^2}{t}\|\widetilde{y}(t-1)\|^2, \qquad (3.38)$$

[4] Note from (3.33) that $\widehat{f}_{k-1}$, $\widehat{g}_{k-1}$, and $\widehat{\delta}_{k-1}$ are functions of (solely) the argument $y(k-1)$. Inequalities (3.36) hold for any random realization of $y(k-1)$, and for any point $x \in \mathbb{R}^d$.

Next, using Assumption 3.6, obtain, $\forall i$:

$$\frac{1}{N}\left(f(x_i(k)) - f^\star\right) \leq \frac{1}{N}\left(f(\overline{x}(k)) - f^\star\right) + \frac{G}{\sqrt{N}}\|\widetilde{x}(k)\|. \tag{3.39}$$

The proof is completed after combining (3.38) and (3.39), taking expectation, and using the bounds on $\mathbb{E}\left[\|\widetilde{x}(k)\|\right] \leq \mathbb{E}\left[\|\widetilde{z}(k)\|\right]$ and $\mathbb{E}\left[\|\widetilde{y}(k)\|^2\right] \leq \mathbb{E}\left[\|\widetilde{z}(k)\|^2\right]$ in Theorem 3.7.

3.5 Algorithm mD–NC

We now present our mD–NC algorithm. Subsection 3.5.1 defines certain additional random matrices needed for representation of mD–NC and presents mD–NC. Subsection 3.5.2 states our result on the convergence rate of mD–NC.

3.5.1 Model and algorithm

This subsection presents our mD–NC algorithm. We continue to consider a sequence of i.i.d. random matrices that obey Assumptions 3.1 and 3.2. For convenience, we introduce here a two-index notation to index these matrices. As we will see, the algorithm D–NC operates in two time scales, i.e., it has the inner iterations, index by s, and the outer iterations, indexed by k. There are $s = 1, 2, ..., \tau_k$ inner iterations at the outer iteration $k = 1, 2, ...$, where τ_k equals:

$$\tau_k = \left\lceil \frac{3\,\log k + \log N}{-\log \overline{\mu}} \right\rceil. \tag{3.40}$$

It can be shown that, for static networks, the term $\log N$ can be dropped. At each inner iteration, nodes utilize one communication round – each node broadcasts a $2d \times 1$ vector to all its neighbors. We denote by $W(k, s)$ the random weight matrix that corresponds to the communication round at the s-th inner iteration and k-th outer iteration. The matrices $W(k, s)$ are ordered in a one-dimensional sequence as $W(k = 1, s = 1), W(k = 1, s = 2), ..., W(k = 1, s = \tau_1), ...W(k = 2, s = 1), ...$ This sequence obeys Assumptions 3.1 and 3.2.

It will be useful to define the products of the weight matrices $W(k, s)$ over an outer iteration k:

$$\mathcal{W}(k) := \Pi_{s=0}^{\tau_k - 1} W(k, \tau_k - s). \tag{3.41}$$

Clearly, the sequence $\{\mathcal{W}(k)\}_{k=1}^{\infty}$ is a sequence of independent (but not identically distributed) matrices.

For future reference, also define $\widetilde{\mathcal{W}}(k) := \mathcal{W}(k) - J$, and:

$$\widetilde{\Psi}(k,t) := \mathcal{W}(k)\mathcal{W}(k-1)...\mathcal{W}(t+1), \tag{3.42}$$

for $t = 0, 1, ..., k-1$.

The Lemma below follows from Assumptions 3.1 and 3.2, independence of the matrices $\mathcal{W}(k)$, the value of τ_k in (3.40), and Lemma 3.3. The Lemma is proved in the Appendix.

Lemma 3.13 Let Assumptions 3.1 and 3.2 hold. Then, for all $k = 1, 2, ...$, for all $s, t \in \{0, 1, ..., k-1\}$:

$$\mathbb{E}\left[\left\|\widetilde{\mathcal{W}}(k)\right\|^2\right] \leq \frac{1}{k^6} \tag{3.43}$$

$$\mathbb{E}\left[\|\widetilde{\Psi}(k,t)\|\right] \leq \frac{1}{k^3(k-1)^3...(t+1)^3} \tag{3.44}$$

$$\mathbb{E}\left[\|\widetilde{\Psi}(k,t)^\top\widetilde{\Psi}(k,t)\|\right] \leq \left(\frac{1}{k^3(k-1)^3...(t+1)^3}\right)^2 \tag{3.45}$$

$$\mathbb{E}\left[\|\widetilde{\Psi}(k,s)^\top\widetilde{\Psi}(k,t)\|\right] \leq \left(\frac{1}{k^3(k-1)^3...(t+1)^3}\right)\left(\frac{1}{k^3(k-1)^3...(s+1)^3}\right). \tag{3.46}$$

The mD–NC algorithm

We now present mD–NC. It uses a constant step-size $\alpha \leq 1/(2L)$. Each node i maintains over (outer iterations) k the solution estimate $x_i(k)$ and an auxiliary variable $y_i(k)$. The mD–NC algorithm is summarized in Algorithm 2. Recall the quantity $\overline{\mu}$ in Lemma 3.3. The inter-node communication occurs only at step 3;

Algorithm 2 The mD–NC algorithm

1: Initialization: Node i sets: $x_i(0) = y_i(0) \in \mathbb{R}^d$; and $k = 1$.

2: Node i calculates:
$$x_i^{(a)}(k) = y_i(k-1) - \alpha\nabla f_i(y_i(k-1)).$$

3: (Consensus) Nodes run average consensus on a $2\,d \times 1$ variable $\chi_i(s,k)$, initialized by $\chi_i(s=0,k) = (x_i^{(a)}(k)^\top, x_i(k-1)^\top)^\top$:
$$\chi_i(s,k) = \sum_{j\in O_i(k)} W_{ij}(k,s)\,\chi_j(s-1,k), \;\; s = 1, 2, ..., \tau_k, \tag{3.47}$$
with τ_k in (3.40), and set $x_i(k) := [\chi_i(s=\tau_k,k)]_{1:d}$ and $x_i^{(b)}(k-1) := [\chi_i(s=\tau_k,k)]_{d+1:2\,d}$. (Here $[a]_{l:m}$ is a selection of l-th, $l+1$-th, ..., m-th entries of vector a.)

4: Node i calculates:
$$y_i(k) = (1+\beta_{k-1})x_i(k) - \beta_{k-1}\,x_i^{(b)}(k-1).$$

5: Set $k \mapsto k+1$ and go to step 2.

step 3 of the k-th outer iteration has τ_k communication rounds. We assume that nodes know beforehand the

constants L, $\overline{\mu}$, and N. We show in Section 3.7 how this requirement can be relaxed.

Vector form. We rewrite mD–NC in vector form, using the definition of matrices $\mathcal{W}(k)$ in (3.41). Use the same compact notation as with mD–NG for $x(k)$, $y(k)$, and $F : \mathbb{R}^{Nd} \to \mathbb{R}^N$. Then:

$$x(k) = (\mathcal{W}(k) \otimes I)\ [y(k-1) - \alpha\, \nabla F(y(k-1))] \tag{3.48}$$

$$y(k) = (1+\beta_{k-1})\, x(k) - \beta_{k-1}\ (\mathcal{W}(k) \otimes I)\ x(k-1), \tag{3.49}$$

$k = 1, 2, ...$, and $x(0) = y(0) \in \mathbb{R}^{Nd}$. Note the formal similarity with mD–NG in (3.10)–(3.11). The differences are that $W(k)$ is replaced with $\mathcal{W}(k)$, and the diminishing step-size $\alpha_k = c/(k+1)$ is replaced with the constant step-size $\alpha_k = \alpha$.

3.5.2 Convergence rate

Define, as with mD–NG (and in the same notation), the disagreements $\widetilde{x}_i(k)$, $\widetilde{y}_i(k)$, $\widetilde{x}(k)$, and $\widetilde{y}(k)$, as well as the augmented vector $\widetilde{z}(k) := \left(\widetilde{y}(k)^\top, \widetilde{x}(k)^\top\right)^\top$. We have the following Theorem on the disagreement bounds.

Theorem 3.14 Consider the mD–NC given in Algorithm 2 under Assumptions 3.1–3.6. Then, for all $k = 1, 2, ...$

$$\mathbb{E}\left[\|\widetilde{z}(k)\|\right] \leq \frac{50\,\alpha\, N^{1/2}\, G}{k^2} \tag{3.50}$$

$$\mathbb{E}\left[\|\widetilde{z}(k)\|^2\right] \leq \frac{50^2\,\alpha^2\, N G^2}{k^4}. \tag{3.51}$$

Theorem 3.15 Consider mD–NC given in Algorithm 2 under Assumptions 3.1–3.6. Let $\|\overline{x}(0) - x^\star\| \leq R$, $R \geq 0$. Then, after

$$\mathcal{K} = \sum_{t=1}^{k} \tau_t \leq \frac{3}{-\log \overline{\mu}}\, [\,(k+1)\log(N(k+1))\,] = O\left(k \log k\right) \tag{3.52}$$

communication rounds, i.e., after k outer iterations, we have, at any node i:

$$\frac{\mathbb{E}\left[f(x_i(k)) - f^\star\right]}{N} \leq \frac{1}{k^2}\left(\frac{2}{\alpha}R^2 + 11\,\alpha^2 L G^2 + \alpha G^2\right), \quad k = 1, 2, ... \tag{3.53}$$

Remark. Theorem 3.15 implies the $O\left(\frac{N^\xi}{(1-\overline{\mu})^2 \mathcal{K}^{2-\xi}}\right)$ convergence rate of mD–NC method in the number of communications $\mathcal{K}$ (in terms of $\frac{\mathbb{E}[f(x_i(k)) - f^\star]}{N}$.) This can be proved analogously to the analysis of D–NC

in Chapter 2.

3.6 Proofs of Theorems 3.14 and 3.15

We now prove the convergence rate results for mD–NC. Subsection 3.6.1 proves Theorem 3.14 and Subsection 3.6.2 proves Theorem 3.15.

3.6.1 Proof of Theorem 3.14

For notational simplicity, we perform the proof for $d = 1$, but the proof extends to a generic $d > 1$. Similarly to the proof of Theorem 3.7, we proceed in three steps. In Step 1, we derive the dynamics for the disagreement $\widetilde{z}(k) = \left(\widetilde{y}(k)^\top, \widetilde{x}(k)^\top\right)$. In Step 2, we unwind the disagreement equation and express $\widetilde{z}(k)$ in terms of the $\widetilde{\Psi}(k,t)$'s in (3.42) and $\mathcal{B}(k,t)$ in (3.18). Finally, Step 3 finalizes the proof using the previously established bounds on the norms of $\widetilde{\Psi}(k,t)$ and $\mathcal{B}(k,t)$.

Step 1. Disagreement dynamics

We write the dynamic equation for $\widetilde{z}(k)$. Recall $B(k)$ in (3.17). Multiplying (3.48)–(3.49) from the left by $(I-J)$, and using $(I-J)\mathcal{W}(k) = \widetilde{\mathcal{W}}(k)(I-J)$, obtain:

$$\widetilde{z}(k) = \Big(B(k) \otimes \widetilde{\mathcal{W}}(k)\Big) \left(\widetilde{z}(k-1) + u'(k-1)\right), \;\; k = 1, 2, ..., \tag{3.54}$$

and $\widetilde{z}(0) = 0$, where

$$u'(k-1) = -\begin{bmatrix} \alpha_{k-1} \nabla F(y(k-1)) \\ 0 \end{bmatrix}. \tag{3.55}$$

Step 2: Unwinding the recursion (3.54)

Recall the products $\mathcal{B}(k,t)$ in (3.18). Then, unwinding (3.54), and using $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$:

$$\widetilde{z}(k) = \sum_{t=0}^{k-1} \Big(\mathcal{B}(k, k-t-2)B(t+1) \otimes \widetilde{\Psi}(k,t)\Big)\, u'(t), \tag{3.56}$$

for all $k = 1, 2, ...$ The quantities $u'(t)$ and $\widetilde{\Psi}(k,t)$ in (3.56) are random, while the $\mathcal{B}(k, k-t-2)$'s are deterministic.

Step 3: Finalizing the proof

Consider $u'(t)$ in (3.55). By Assumption 3.6, we have $\|\nabla F(y(t))\| \leq \sqrt{N}G$. Using the latter, we obtain that $\|u'(t)\| \leq \alpha\sqrt{N}\,G$, for any random realization of $u'(t)$. We use the latter bound, Lemma 3.12, the sub-multiplicative and sub-additive properties of norms, and $\|B(t+1)\| \leq 3$, $\forall t$; from (3.56):

$$\|\widetilde{z}(k)\| \leq 3\left(8\,\alpha\sqrt{N}\,G\right)\sum_{t=0}^{k-1}\|\widetilde{\Psi}(k,t)\|\,\frac{(k-t-1)(t+1)}{k} + 3\left(5\,\alpha\sqrt{N}\,G\right)\sum_{t=0}^{k-1}\|\widetilde{\Psi}(k,t)\|.$$

Taking expectation, using $\frac{(k-t-1)(t+1)}{k} \leq t+1$, and using Lemma 3.13:

$$\begin{aligned}
\mathbb{E}\left[\|\widetilde{z}(k)\|\right] &\leq 3\left(8\,\alpha\sqrt{N}\,G\right)\sum_{t=0}^{k-1}\frac{1}{k^3(k-1)^3...(t+2)^3(t+1)^2} \\
&+ 3\left(5\,\alpha\sqrt{N}\,G\right)\sum_{t=0}^{k-1}\frac{1}{k^3(k-1)^3...(t+2)^3(t+1)^3} \\
&\leq 3\left(8\,\alpha\sqrt{N}\,G\right)\frac{1}{k^2} \\
&+ 3\left(5\,\alpha\sqrt{N}\,G\right)\frac{1}{k^2} \leq \frac{50\,\alpha\sqrt{N}\,G}{k^2}.
\end{aligned}$$

Thus, the result in (3.50).

We now prove (3.51). Consider $\|\widetilde{z}(k)\|^2$. From (3.56), we obtain:

$$\begin{aligned}
\|\widetilde{z}(k)\|^2 &= \sum_{t=0}^{k-1}\sum_{s=0}^{k-1}u'(t)^\top\left(B(t+1)^\top\mathcal{B}(k,k-t-2)^\top\otimes\widetilde{\Psi}(k,t)^\top\right) \\
&\times \left(\mathcal{B}(k,k-t-2)B(t+1)\otimes\widetilde{\Psi}(k,s)\right)u'(s) \\
&= \sum_{t=0}^{k-1}\sum_{s=0}^{k-1}u'(t)^\top\left(B(t+1)^\top\mathcal{B}(k,k-t-2)^\top\right. \\
&\times \left.\mathcal{B}(k,k-s-2)B(t+1)\right)\otimes\left(\widetilde{\Psi}(k,t)^\top\,\widetilde{\Psi}(k,s)\right)u'(s),
\end{aligned}$$

where the last inequality uses the Kronecker product property $(A\otimes B)(C\otimes D) = (AC)\otimes(BD)$. Next, by the sub-additive and sub-multiplicative properties of norms, and $\|B(t+1)\| \leq 3$, $\forall t$, obtain:

$$\begin{aligned}
\|\widetilde{z}(k)\|^2 &\leq 9\sum_{t=0}^{k-1}\sum_{s=0}^{k-1}\|\mathcal{B}(k,k-t-2)\|\,\|\mathcal{B}(k,k-s-2)\|\left\|\widetilde{\Psi}(k,t)^\top\,\widetilde{\Psi}(k,s)\right\|\,\|u'(t)\|\,\|u'(s)\| \qquad (3.57)\\
&\leq 9\sum_{t=0}^{k-1}\sum_{s=0}^{k-1}(8(t+1)+5)\,(8(s+1)+5)\left\|\widetilde{\Psi}(k,t)^\top\,\widetilde{\Psi}(k,s)\right\|\,\alpha^2\,NG^2,
\end{aligned}$$

where the last inequality uses $(k-s-1)(s+1)/k \leq s+1$, Lemma 3.12 and $\|u(t)\| \leq \left(\alpha\sqrt{N}G\right)/(t+1)$. Taking expectation and applying Lemma 3.13, we obtain:

$$\begin{aligned}
\mathbb{E}\left[\|\widetilde{z}(k)\|^2\right] &\leq 9\left(\alpha^2 NG^2\right)\sum_{t=0}^{k-1}\sum_{s=0}^{k-1}(8(t+1)+5)\ (8(s+1)+5) \\
&\times \left(\frac{1}{k^3...(t+1)^3}\right)\left(\frac{1}{k^3...(s+1)^3}\right) \\
&= 9\left(\alpha^2 NG^2\right)\left(\sum_{t=0}^{k-1}(8(t+1)+5)\,\frac{1}{k^3...(t+1)^3}\right)^2 \\
&\leq \frac{50^2\,\alpha^2\,NG^2}{k^4}.
\end{aligned}$$

Thus, the bound in (3.51). The proof of Theorem 3.14 is complete.

3.6.2 Proof outline of Theorem 3.15

The proof is similar to the proof of Theorem 8 in [9] (version v2), and hence we give only an outline. Consider the global averages $\overline{x}(k)$ and $\overline{y}(k)$, defined analogously to the case of mD–NG. Then, $\overline{x}(k)$ and $\overline{y}(k)$ evolve according to (3.34)–(3.35), with $L_{k-1} := N/\alpha$, and $\widehat{g}_{k-1}$ defined as in (3.33). Also, inequalities (3.36) hold with $L_{k-1} := N/\alpha$ and $\widehat{f}_{k-1}$ and $\widehat{g}_{k-1}$ defined as in (3.33). We can thus apply Lemma 2.5 in Chapter 2, which gives:

$$\frac{1}{N}\left(f(\overline{x}(k)) - f^\star\right) \leq \frac{2}{\alpha\,k^2}\|\overline{x}(0) - x^\star\|^2 + \frac{L}{N\,k^2}\sum_{t=1}^{k}\|\widetilde{y}(t-1)\|^2(t+1)^2.$$

(Compare the last equation with (3.38).) The remainder of the proof proceeds analogously to the case of Theorem 3.8.

3.7 Discussion and extensions

This section discusses extensions and corollaries of our analysis. We show that: 1) the requirement on the a priori knowledge of $L, \overline{\mu}$, and N can be relaxed with both mD–NG and mD–NC; 2) establish rates in the convergence in probability of mD–NG and mD–NC; 3) show almost sure convergence with mD–NC; and 4) establish a convergence rate in the second moment with both methods.

Relaxing knowledge of $L, \overline{\mu}$, and N

The mD–NG method requires only the knowledge of L to set the step-size $\alpha_k = c/(k+1)$, $c \leq 1/(2L)$. We demonstrate that the rate $O(\log k/k)$ (with a deteriorated constant) still holds if nodes use arbitrary $c > 0$. Initialize all nodes to $x_i(0) = y_i(0) = 0$, suppose that $c > 1/(2L)$, and let $k' = 2\,c\,L$. Applying Lemma 2.5, as in the proof of Theorem 2.8 (b), for all $k > k'$, surely:

$$\begin{aligned} &\tfrac{(k+1)^2-1}{k+1} \quad (f(\overline{x}(k)) - f^\star) \\ &\leq \qquad k' \left(f(\overline{x}(k'-1)) - f^\star\right) + \frac{2N}{c}\left(2\left\|\overline{v}(k'-1)\right\|^2 + 2\left\|x^\star\right\|^2\right) + \sum_{t=1}^{k} \frac{(t+1)^2}{t} L \|\widetilde{y}(t-1)\|^2. \end{aligned} \tag{3.58}$$

Further, consulting the proof of Theorem 2.8 (b), surely:

$$\|\overline{v}(k'-1)\|^2 \leq (2k'+1)^2 \left(3^{k'}\right)^2 2\,c\,G. \tag{3.59}$$

Finally, Theorem 3.7 holds unchanged for $c > 1/(2L)$, and thus $\sum_{t=1}^{k} \frac{(t+1)^2}{t} L \mathbb{E}\left[\|\widetilde{y}(t-1)\|^2\right] = O(\log k)$. Multiplying (3.58) by $\frac{k+1}{(k+1)^2-1}$, taking expectation on the resulting inequality, and applying Theorem 3.7, we obtain the $O(\log k/k)$ rate, as desired.

The mD–NC algorithm uses the constant step-size $\alpha \leq 1/(2L)$ and τ_k in (3.40). We adapt mD–NC to avoid the use of $L, \overline{\mu}$, and N, by setting: 1) a diminishing step-size $\alpha_k = 1/k^p$, $p \in (0, 1]$; and 2) $\tau_k = k$ (as suggested in [29]). We demonstrate that the adapted mD–NC achieves rate $O(1/k^{2-p})$. Let $k'' = (2L)^{1/p}$. Then, by Lemma 2.5, for all $k \geq k''$, surely:

$$\begin{aligned} &\tfrac{(k+1)^2-1}{(k+1)^p} \quad (f(\overline{x}(k)) - f^\star) \\ &\leq \qquad (k')^{2-p}\left(f(\overline{x}(k'-1)) - f^\star\right) + 2N\,\left(2\left\|\overline{v}(k'-1)\right\|^2 + 2\left\|x^\star\right\|^2\right) + \sum_{t=1}^{k} \frac{(t+1)^2}{t^p} L \|\widetilde{y}(t-1)\|^2. \end{aligned} \tag{3.60}$$

Further, (3.59) holds here as well (surely.) We now modify the argument on the sum in (3.60). By Lemma 3.13 and the value $\tau_k = k$, we have: $\mathbb{E}\left[\|\widetilde{\mathcal{W}}(k)\|^2\right] \leq N^2 \overline{\mu}^{2\,k}$. From this equality, $\forall k \geq k''' := \left(\frac{6(\log N+1)}{-\log \overline{\mu}}\right)^2$: $\mathbb{E}\left[\|\widetilde{\mathcal{W}}(k)\|^2\right] \leq \frac{1}{k^4}$. Next, consider the matrix

$$\widetilde{\Psi}(k,s)^\top \widetilde{\Psi}(k,t) = \left(\widetilde{\mathcal{W}}(k)...\widetilde{\mathcal{W}}(s+1)\right)^\top \left(\widetilde{\mathcal{W}}(k)...\widetilde{\mathcal{W}}(t+1)\right),$$

for arbitrary $k \geq k'''$, and arbitrary $s, t \in \{0, 1, ..., k-1\}$. Clearly,

$$\left\| \widetilde{\Psi}(k,s)^\top \widetilde{\Psi}(k,t) \right\| \leq \left\| \widetilde{\mathcal{W}}(k) \right\|^2,$$

and hence,

$$\mathbb{E}\left[\left\| \widetilde{\Psi}(k,s)^\top \widetilde{\Psi}(k,t) \right\| \right] \leq \frac{1}{k^4}, \ \ \forall s, t \in \{0, 1, ..., k-1\}, \ \ \forall k \geq k'''.$$

Now, from step 3 of the proof of Theorem 3.14, the above implies: $\mathbb{E}\left[\|\widetilde{y}(k)\|^2\right] \leq \mathbb{E}\left[\|\widetilde{z}(k)\|^2\right] \leq \frac{C}{k^4}$, for all $k \geq k'''$, where $C > 0$ is independent of k. Hence, we obtain the desired bound on the sum:

$$\sum_{t=1}^{\infty} \frac{(t+1)^2}{t^p} L \mathbb{E}\left[\|\widetilde{y}(t-1)\|^2\right] = O(1).$$

Using the latter, (3.59), multiplying (3.60) by $\frac{(k+1)^p}{(k+1)^2-1}$, and taking expectation in (3.60), we obtain the desired rate $O(1/k^{2-p})$.

Convergence in probability and almost sure convergence

Through the Markov inequality, Theorems 3.8 and 3.15 imply, for any $\epsilon > 0$, the following:

$$\begin{aligned}
&\mathrm{mD-NG}: \ \mathbb{P}\left(k^{1-\xi} \ (f(x_i(k)) - f^\star) > \epsilon \right) \to 0 \ \text{ as } \ k \to \infty, \ \ \forall i \\
&\mathrm{mD-NC}: \ \mathbb{P}\left(k^{2-\xi} \ (f(x_i(k)) - f^\star) > \epsilon \right) \to 0 \ \text{ as } \ k \to \infty, \ \ \forall i,
\end{aligned}$$

where $\xi > 0$ is arbitrarily small. Furthermore, by the arguments in, e.g., ([80], Subsection IV–A), with mD–NC, we have that, $\forall i$, $f(x_i(k)) - f^\star \to 0$, almost surely.

Convergence rates in the second moment

Consider a special case of the random network model in Assumptions 3.1 and 3.2. Define the random graph $G(k)$ to be the graph that supports the nonzero pattern of the random matrix $W(k)$; that is, $G(k) = (\mathcal{N}, E)$, with $E = \{\{i, j\} : \ W_{ij}(k) > 0, \ i < j\}$. In other words, $G(k)$ is the graph that supports *a random realization* of matrix $W(k)$. We assume that $G(k)$ is connected with positive probability. This Assumption holds, e.g., with spatio-temporally independent link failures, but it does not hold, e.g., with pairwise gossip. (With gossip, only one edge occurs at a time, and hence all realizations of $G(k)$ are disconnected.) Under

the latter Assumption, we establish the following bounds on the second moment of the optimality gaps:

$$\text{mD}-\text{NG}: \ \mathbb{E}\left[(f(x_i(k))-f^\star)^2\right] = O\left(\frac{\log^2 k}{k^2}\right), \ \ \forall i, \tag{3.61}$$

$$\text{mD}-\text{NC}: \ \mathbb{E}\left[(f(x_i(k))-f^\star)^2\right] = O\left(\frac{1}{k^4}\right), \ \forall i, \tag{3.62}$$

where (3.62) holds for mD–NC with a modified value of τ_k. (See the Appendix for details.) We interpret (3.61), while (3.62) is similar. The result (3.61) shows that, not only the mean of the optimality gap decays as $O(\log k/k)$ (by Theorem 3.8), but also the standard deviation about the mean is of order $O(\log k/k)$. The proof of (3.61)–(3.62) is in the Appendix.

3.8 Simulation example

We provide a simulation example that corroborates convergence rates of the proposed mD–NG and mD–NC in the presence of link failures, as well as their rates faster than the rates of the method in [2]. We also compare D–NG and mD–NC with the original variants in Chapter 2.

Setup

We consider a $N = 10$ node network and the Huber loss cost functions, which arise, e.g., in distributed robust estimation in sensor networks [4]; the function $f_i : \mathbb{R} \to \mathbb{R}$ is $f_i(x) = \|x - \theta_i\|^2$ if $\|x - \theta_i\| \leq 1$, and $f_i(x) = \|x - \theta_i\| - 1/2$, else, $\theta_i \in \mathbb{R}$. (The f_i's obey Assumptions 3.4 and 3.6.) We generate θ_i's as follows. For $i = 1, 2, 3$, we set $\theta_i = \theta^\bullet(1 + \nu_i)$, where $\theta^\bullet = 4$, and ν_i is generated randomly from the uniform distribution on $[-0.1, 0.1]$. For $j = 4, 5, ..., 10$, we set $\theta_j = (-\theta^\bullet)(1 + \nu_j)$, where ν_j is generated from the same uniform distribution.

The network model has a connected supergraph $\mathcal{G} = (\mathcal{N}, E)$ with 26 links. It is generated randomly by placing nodes at random on a unit 2D square and connecting the pairs of nodes whose distance is less than a prescribed radius. We consider two scenarios: 1) random network (failing links), and 2) static network. With the random network, each link $\{i, j\} \in E$ fails independently in time and independently from other links, with probability 0.9. When a link $\{i, j\} \in E$ is online, we set the weights $W_{ij}(k) = W_{ji}(k) = 1/N = 1/10$. (We set $W_{ii}(k) = 1 - \sum_{j \in O_i(k) - \{i\}} W_{ij}(k)$, $\forall i$.) With the static network, we consider the same supergraph $\mathcal{G}$, and, $\forall \{i, j\} \in E$, we set $W_{ij} = W_{ji} = 1/N$. With D–NG and mD–NG, we set the step-size $\alpha_k = 1/(k + 1)$, while with D–NC and mD–NC, we use $\alpha = 1/2$. With the algorithm in [2], we use the step-size $\alpha_k = 1/\sqrt{k}$. With random networks, for both variants of D–NC, we set τ_k as in (3.40);

with static networks, we use $\tau_k = \left\lceil \frac{3 \log k}{-\log \overline{\mu}} \right\rceil$. (As indicated in Section 3.5, the $\log N$ term is not needed with static networks.)

With both scenarios, we simulate the mD–NG and mD–NC, the original variants D–NG and D–NC in Chapter 2, and the method in [2]. We initialize all the methods to $x_i(0) = y_i(0) = 0$, $\forall i$. We generate one sample path (simulation run), and we estimate the average normalized optimality gap $\mathbf{err_f} = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i) - f^\star}{f(0) - f^\star}$ versus the total number $\mathcal{K}'$ of scalar transmissions, cumulatively across all nodes. We count both the successful and unsuccessful (due to link failures) transmissions. All our plots below are in $\log_{10} - \log_{10}$ scale.

Results: Scenario with link failures

.

Figure 3.1 (top left) shows the performance of mD–NG, mD–NC, D–NC, and the method in [2]. We can see that all methods exhibit convergence in the presence of (severe) link failures. With respect to convergence rates, we can see, for example, that the mD–NG shows improvement over [2]. Note the better (larger) slope of decay with the mD–NG with respect to [2]. Further, Figure 3.1 (top right) shows that the (original) D–NG method diverges on this example.

Results: Static network scenario

Figure 3.1 (bottom left) shows the performance of the same methods on static network. Note again a faster rate (better slope) of the mD–NG and D–NG's with respect to [22]. Further, we can see that the original D–NG method performs better than mD–NG. Hence, the modified method looses slightly in performance, but gains robustness to link failures. The D–NC and mD–NC methods perform the same, both on static and on random networks.

As noted earlier, the original D–NG method in 2 requires the weight matrix W to be positive definite. In certain scenarios, nodes may not be able to check whether a given W is positive definite or not. We include a simulation that compares D–NG and mD–NG's by accounting for this effect. (The network is assumed static here.) We take the Metropolis weight matrix W (see [65]) which is not positive definite. As the original D–NG requires a positive definite W, we replace W with $W' = \frac{1.01}{2} I + \frac{0.99}{2} W$, which is positive definite, but has a larger (worse) $\overline{\mu}$. With mD–NG, we use the (non-positive definite) W. The remaining system parameters are the same as in the previous example. Figure 3.1 (bottom right) plots $\mathbf{err_f}$ versus the total number of scalar transmissions $\mathcal{K}'$ for the original D–NG (red, dotted line), and mD–NG (green, solid line). We can see that the methods perform almost the same, with the original D–NG performing slightly better.

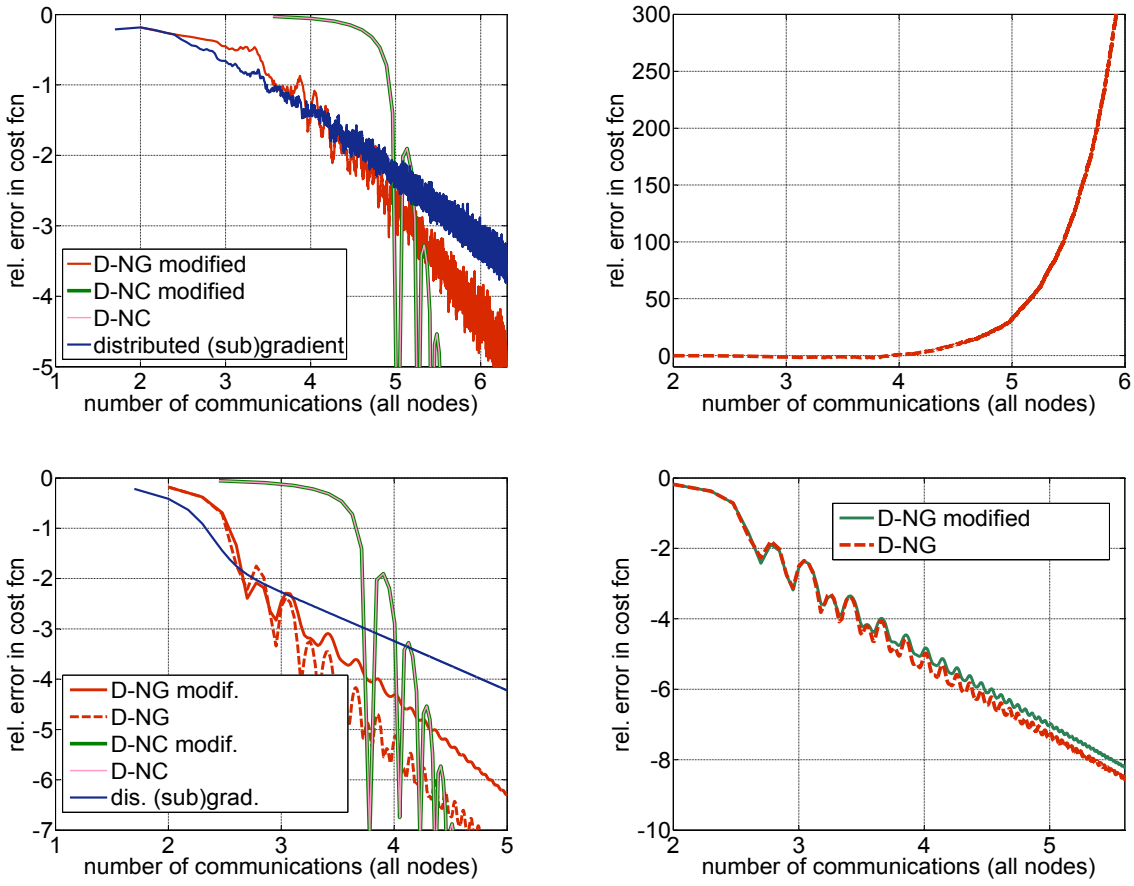

Figure 3.1: Average normalized optimality gap $\mathbf{err_f} = \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)-f^\star}{f(0)-f^\star}$ versus the total number $\mathcal{K}'$ of scalar transmissions, cumulatively across all nodes (in the $\log_{10}-\log_{10}$ scale.) Top (left and right): Scenario with link failures; Bottom left: Static network scenario. Bottom right: Static network scenario; comparison of D–NG and mD–NG when the weight matrix W is not positive definite.

3.9 Conclusion

In this Chapter, we modified our D–NG and D–NC methods to operate under randomly varying networks, modeled by a sequence $\{W(k)\}$ of independent, identically distributed random stochastic matrices. The f_i's are convex and have Lipschitz continuous and bounded gradients. We establish convergence rates for the two modified methods, termed mD–NG and mD–NC, in terms of the expected optimality gap at the cost function at arbitrary node i. The mD–NG algorithm achieves rates $O(\log k/k)$ and $O(\log \mathcal{K}/\mathcal{K})$, where k is the number of per-node gradient evaluations and $\mathcal{K}$ is the number of per-node communications. The mD–NC algorithm has rates $O(1/k^2)$ and $O(1/\mathcal{K}^{2-\xi})$, with $\xi > 0$ arbitrarily small. Simulation examples on the networks with link failures and Huber loss functions illustrate our findings.

Chapter 4

Distributed Nesterov-like Gradient Methods: Alternative Function Classes

4.1 Introduction

In Chapters 2 and 3, we established convergence rates for D–NG and D–NC methods, as well as their modified variants, for unconstrained optimization when the costs f_i's are convex, differentiable, and have Lipschitz continuous and bounded gradients. In this Chapter, we analyze our methods for alternative function classes, including also constrained optimization where each node has the same constraint set $\mathcal{X}$. We still consider differentiable convex costs with Lipschitz continuous derivative, but we do not explicitly require that the gradients be bounded. Hence, we enlarge the class of costs f_i's that are applicable to our methods. For example, we can include quadratic costs, as well as the costs that arise with source localization problems (see Section 4.7.) Furthermore, we allow for constraints, thus covering, e.g., model predictive control problems, e.g., [36].

Chapter outline. We outline the chapter. Section 4.2 briefly introduces the general setup. Section 4.3 analyzes D–NG for unconstrained optimization when the bounded gradients Assumption is replaced with a certain growth condition. We show that the method achieves rate $O(\log k/k)$ and $O(\log \mathcal{K}/\mathcal{K})$ in the number of per-node gradient evaluations k and the number of per-node communications $\mathcal{K}$. Section 4.4 presents the projected D–NG method with constant step-size for constrained problems. The method converges to an ϵ-neighborhood of the optimal cost after $O(1/\epsilon)$ per-node communications and $O(1/\epsilon)$ per-node gradient evaluations. Section 4.5 presents the projected mD–NC method for constrained problems. We show that the method converges at rates $O(1/k^2)$ and $O(1/\mathcal{K}^{2-\xi})$. Sections 4.3 and 4.4 consider static networks, while

Section 4.5 allows for random networks as well. Section 4.6 proves a technical result on an inexact projected Nesterov gradient method in Section 4.5. Section 4.7 provides a simulation example on the acoustic source localization problem. Finally, Section 4.8 gives a closing discussion.

Technically, Sections 4.3 and 4.4 bring an approach to the analysis of D–NG that is different from the approach in Chapters 2 and 3. Namely, we introduce certain auxiliary functions Ψ, that we refer to as the clone functions (see ahead (4.11)). We show that the D–NG method is the (centralized) Nesterov gradient method on the clone function Ψ. This allows us to indirectly prove in Section 4.3 that the gradients $\nabla f_i(y_i(k))$ along iterations are bounded, without requiring that the gradients $\nabla f_i(x)$ are uniformly bounded over the whole space $\mathbb{R}^d$. Actually, the clone approach yields tighter upper bounds than the approach in Chapters 2 and 3 *under a constant step-size*. (See Section 4.4 for details.)

Notation. Throughout the Chapter, we denote by: $\mathbb{R}^d$ the d-dimensional real coordinate space, $d \geq 1$; A_{ij} the entry in the i-th row and j-th column of a matrix A; a_i the i-th entry of a vector a; $(\cdot)^\top$ the transpose; $\|\cdot\| = \|\cdot\|_2$ the Euclidean (respectively, spectral) norm of its vector (respectively, matrix) argument (We note that $\|\cdot\|$ also denotes the modulus of a scalar throughout); $\lambda_i(\cdot)$ the i-th smallest eigenvalue; $|\cdot|$ the cardinality of a set; $\nabla \mathcal{J}(y)$ the gradient evaluated at y of a function $\mathcal{J} : \mathbb{R}^d \to \mathbb{R}$, $d \geq 1$. Finally, for positive sequences η_n and χ_n, we have: $\eta_n = O(\chi_n)$ if $\limsup_{n\to\infty} \frac{\eta_n}{\chi_n} < \infty$.

Parts of the material in Chapter 4 have been published in [49, 50].

4.2 Setup

We briefly outline the general setup, and we later add details for each of our specific studies. We consider the constrained problem:

$$\text{minimize} \ \sum_{i=1}^{N} f_i(x) =: f(x) \ \text{ subject to } \ x \in \mathcal{X}, \tag{4.1}$$

where $f_i : \mathbb{R}^d \to \mathbb{R}$ is node i's locally known convex cost function, and $\mathcal{X} \subset \mathbb{R}^d$ is a closed, convex constrained set known by all nodes. We specify further Assumptions on the f_i's and $\mathcal{X}$ in each of the Sections 4.3, 4.4, and 4.5. We associate with problem (4.1) a network $\mathcal{V}$ of N nodes, described by the graph $\mathcal{G} = (\mathcal{V}, E)$, where $E \subset \mathcal{V} \times \mathcal{V}$ is the set of edges, and we require the following.

Assumption 4.1 The graph $\mathcal{G} = (\mathcal{V}, E)$ is connected, undirected, and simple (no self/multiple links).

We recall the unweighted, symmetric graph Laplacian matrix $\mathcal{L}$ by: 1) $\mathcal{L}_{ij} = -1$, if $\{i,j\} \in E$, $i \neq j$; 2) $\mathcal{L}_{ij} = 0$, if $\{i,j\} \notin E$, $i \neq j$; and 3) $\mathcal{L}_{ii} := -\sum_{j \neq i} \mathcal{L}_{ij}$, $\forall i$. Because the graph is connected, we have that the algebraic connectivity $\lambda_2(\mathcal{L}) > 0$. Further, denote by O_i the neighborhood set of node i (including i), and ℓ_i the degree (number of neighbors) of node i.

4.3 D–NG method: Growth assumption

In this Section, we consider D–NG for unconstrained optimization, when the f_i's satisfy a growth condition. Subsection 4.3.1 details the Assumptions and setup, Subsection 4.3.2 defines clone functions Ψ and gives their properties, and Subsection 4.3.3 performs convergence rate analysis. Throughout the current section, we consider static networks.

4.3.1 Assumptions and setup

We consider the unconstrained version of (4.1) with $\mathcal{X} \equiv \mathbb{R}^d$, and we impose the following two Assumptions on the f_i's.

Assumption 4.2 For all i, $f_i : \mathbb{R}^d \to \mathbb{R}$ is convex, differentiable, and has Lipschitz continuous derivative with constant L, i.e., for all i:

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L \|x - y\|, \ \ \forall x, y \in \mathbb{R}^d.$$

Assumption 4.3 (Growth assumption) There exist two positive scalars b and B, such that, for all i, $f_i(x) \geq b \|x\|$ whenever $\|x\| \geq B$.

Assumption 4.2 is standard in the analysis of gradient methods. Assumption 4.3 says that the function grows at least as $b\|x\|$ when $\|x\|$ is sufficiently large. The two Assumptions hold with many costs, e.g., quadratics with positive Hessians, and source localization costs (see Section 4.7). Under Assumption 4.3, each f_i is coercive[1], and so is $f := \sum_{i=1}^N f_i$. Thus, there exist $x_i^\star$, $i = 1, ..., N$, and $x^\star$, such that $f_i^\star := \inf_{x \in \mathbb{R}^d} f_i(x) = f_i(x_i^\star)$, and $f^\star := \inf_{x \in \mathbb{R}^d} f(x) = f(x^\star)$. Without loss of generality (w.l.o.g.), we choose the constant B in Assumption 4.3 such that:

$$f_i^\star \ < \ b\,B, \ \ \forall i = 1, ..., N, \tag{4.2}$$

$$f^\star \ < \ N\,b\,B. \tag{4.3}$$

[1]Coercive means that $f_i(x) \to +\infty$ whenever $\|x\| \to +\infty$.

Hence, any minimizer $x_i^\star$ of f_i, $\forall i$, and any minimizer $x^\star$ of f, belongs to the closed ball $\{x \in \mathbb{R}^d : \|x\| \leq B\}$.

Introduce the function $F : \mathbb{R}^N \to \mathbb{R}$, as:

$$F(x) = F(x_1, ..., x_N) = f_1(x_1) + ... + f_N(x_N). \tag{4.4}$$

For future reference, we introduce the following sub-level set:

$$\mathcal{S} := \left\{ x \in \mathbb{R}^N : \; F(x) = \sum_{i=1}^{N} f_i(x_i) \leq N\, b\, B \right\}, \tag{4.5}$$

and the following two constants:

$$\begin{aligned} \mathcal{D} &:= \sup_{x \in \mathcal{S}} \|x\| < \infty \\ G &:= \sup_{x \in \mathcal{S}} \|\nabla F(x)\| < \infty. \end{aligned} \tag{4.6}$$

The set $\mathcal{S}$ is compact, because the function F is coercive by Assumption 4.3. The two suprema in (4.6) are attained at some points and are finite, because the set $\mathcal{S}$ is compact, and the functions $\|\cdot\|$ and $\|\nabla F(\cdot)\|$ are continuous (The latter function is continuous due to continuity of the gradients of the f_i's – see Assumption 4.2.)

We consider the D–NG algorithm in Chapter 2; we briefly recall it here, for convenience. Each node i updates its solution estimate $x_i(k)$ and an auxiliary variable $y_i(k)$ over iterations k as follows[2]:

$$x_i(k) = (1 - \ell_i\, w)\, y_i(k-1) + w \sum_{j \in O_i - \{i\}} y_j(k-1) - \alpha \nabla f_i(y_i(k-1)) \tag{4.7}$$

$$y_i(k) = x_i(k) + \beta_{k-1}\, (x_i(k) - x_i(k-1)), \;\; k = 1, 2, ..., \tag{4.8}$$

with $x_i(0) = y_i(0) \in \mathbb{R}^d$. The step-size α_k and the sequence β_k are:

$$\alpha_k = \frac{c}{k+1}, \quad \beta_k = \frac{k}{k+3}, \;\; k = 0, 1, ... \tag{4.9}$$

[2]We assign equal weights w_0 to all neighbors; generalization to unequal weights is straightforward.

We choose c and w as:

$$\begin{aligned} c &\leq \frac{1}{\rho\,\lambda_N(\mathcal{L}) + L} \\ w &= c\,\rho. \end{aligned} \tag{4.10}$$

Here, ρ is a positive constant; e.g., it can be set to $\rho = 1$. To satisfy (4.10), we require that nodes know beforehand (upper bounds on) ρ, $\lambda_N(\mathcal{L})$, and L. We can set $\rho = 1$. It can be shown that $\lambda_N(\mathcal{L}) \leq 2\,\max_{i=1,...,N}\ell_i$. If ∇f_i has a Lipschitz constant L_i, known by node i, we can take L as $L := \max_{i=1,...,N} L_i$. Hence, requirement (4.10) is accomplished beforehand through two distributed maximum computations – in $O(N)$ per-node scalar communications. (See also Chapter 2 for a similar discussion.)

Here, and throughout the whole Chapter, to avoid notational clutter, we assume equal initialization $\overline{x}(0) := x_i(0) = y_i(0) = x_j(0) = y_j(0)$, $\forall i, j$, and we let $d = 1$, but the results extend for a generic d as well.

4.3.2 Clone functions Ψ_k

For a real number $k \geq 0$, consider the following (clone) unconstrained optimization problem over $x = (x_1, x_2, ..., x_N) \in \mathbb{R}^N$:

$$\text{minimize} \quad \Psi_k(x) := \textstyle\sum_{i=1}^N f_i(x_i) + \frac{k\rho}{2}\,x^\top \mathcal{L} x, \;\; \rho > 0. \tag{4.11}$$

In (4.11), recall that $\mathcal{L}$ is the graph Laplacian matrix. Recall the step-size $\alpha_{k-1} = c/k$, and let $w = c\rho$. Introduce compact notation for the nodes' estimates $x(k) := (x_1(k), ..., x_N(k))^\top$, and $y(k) := (y_1(k), ..., y_N(k))^\top$. Then, it is easy to verify that algorithm (4.7)–(4.8) can be re-written as:

$$x(k) = y(k-1) - \alpha_{k-1}\nabla\Psi_k(y(k-1)) \tag{4.12}$$

$$y(k) = x(k) + \beta_{k-1}\left(x(k) - x(k-1)\right),\; k = 1, 2, ... \tag{4.13}$$

with α_k and β_k in (4.9) and the initialization is $x(0) = y(0) = \overline{x}(0)\mathbf{1}$. Hence, at iteration k, algorithm (4.7)–(4.8) performs the (exact) Nesterov gradient step with respect to the clone function Ψ_k.

We impose that the step-size satisfies:

$$\alpha_{k-1} = c/k \leq 1/L_{\Psi_k},$$

where L_{Ψ_k} is a Lipschitz constant of the gradient of Ψ_k, which we can take as:

$$L_{\Psi_k} = k\ (\rho\,\lambda_N(\mathcal{L}) + L)\,.$$

Thus, we can choose c as in (4.10).

Properties of the clone functions Ψ_k

The next Lemma states certain properties of the clone functions Ψ_k's and problem (4.11). The Lemma also relates (4.11) with the original problem (3.5).

Lemma 4.4 (Properties of (4.11)*)* Consider (4.11), and let Assumption 4.1 hold. Then:

(a) There exists a solution $x^{\mathrm{c}}(k) = (x_1^{\mathrm{c}}(k), ..., x_N^{\mathrm{c}}(k))$ to (4.11) that satisfies $\inf_{x\in\mathbb{R}^N} \Psi_k(x) = \Psi(x^{\mathrm{c}}(k)) =: \Psi_k^\star > -\infty$. Further, the corresponding solution set is compact.

(b) $f^\star \geq \Psi_k^\star \geq \Psi_t^\star$ for all k, t, $k > t$.

(c) For any $x^{\mathrm{c}}(k)$, there holds: $\|\nabla f_i(x_{\mathrm{avg}}^{\mathrm{c}}(k))\| \leq L\mathcal{D} + G =: G^\star$, for all i, k, where $x_{\mathrm{avg}}^{\mathrm{c}}(k) = \frac{1}{N}\sum_{i=1}^N x_i^{\mathrm{c}}(k)$.

(d) For any $k > 0$, $\Psi_k^\star \geq f^\star - \frac{N(G^\star)^2}{2\rho k \lambda_2(\mathcal{L})}$.

Proof: For part (a), note that the function Ψ_k is, by Assumption 4.3, coercive. Also, the function Ψ_k is closed and convex. Hence, as Ψ_k is closed, convex, and coercive, problem (4.11) is solvable, the solution set is compact, and $\Psi_k^\star > -\infty$ (see, e.g., [81]).

We prove part (b). Fix some k, and note that:

$$f^\star = \Psi_k(x^\star\mathbf{1}) = \sum_{i=1}^N f_i(x^\star) \geq \Psi_k(x^{\mathrm{c}}(k)) = \Psi_k^\star,$$

and thus $f^\star \geq \Psi_k^\star$. Next, fix k, t, $k > t$, and note that:

$$\begin{aligned}\Psi_t^\star &= \Psi_t(x^{\mathrm{c}}(t)) \leq \Psi_t(x^{\mathrm{c}}(k)) = \sum_{i=1}^N f_i(x_i^{\mathrm{c}}(k)) + \frac{\rho\,t}{2} x^{\mathrm{c}}(k)^\top \mathcal{L} x^{\mathrm{c}}(k) \\ &\leq \sum_{i=1}^N f_i(x_i^{\mathrm{c}}(k)) + \frac{\rho\,k}{2} x^{\mathrm{c}}(k)^\top \mathcal{L} x^{\mathrm{c}}(k) = \Psi_k^\star,\end{aligned}$$

and so $\Psi_t^\star \leq \Psi_k^\star$ whenever $t < k$, which completes the proof of part (b).

We now prove part (c). From part (b):

$$F(x^{\mathrm{c}}(k)) = \sum_{i=1}^{N} f_i(x_i^{\mathrm{c}}(k)) \leq \Psi_k^\star \leq f^\star \leq N\, b\, B,\, \forall k.$$

Thus, $x^{\mathrm{c}}(k)$ belongs to set $\mathcal{S}$, and then, in view of (4.6), we have: $\|x_i^{\mathrm{c}}(k)\| \leq \|x^{\mathrm{c}}(k)\| \leq \mathcal{D}$, $\forall i$, $\forall k$. Further:

$$\|x_{\mathrm{avg}}^{\mathrm{c}}(k)\| = \left\| \frac{1}{N} \sum_{i=1}^{N} x_i^{\mathrm{c}}(k) \right\| \leq \frac{1}{N} N \|x^{\mathrm{c}}(k)\| \leq \mathcal{D}.$$

We now upper bound $\|\nabla f_i(x_{\mathrm{avg}}^{\mathrm{c}}(k))\|$:

$$\begin{aligned}
\|\nabla f_i(x_{\mathrm{avg}}^{\mathrm{c}}(k))\| &= \|\nabla f_i(x_{\mathrm{avg}}^{\mathrm{c}}(k)) - \nabla f_i(x_i^{\mathrm{c}}(k)) + \nabla f_i(x_i^{\mathrm{c}}(k))\| \\
&\leq \|\nabla f_i(x_{\mathrm{avg}}^{\mathrm{c}}(k)) - \nabla f_i(x_i^{\mathrm{c}}(k))\| + \|\nabla f_i(x_i^{\mathrm{c}}(k))\| \\
&\leq L\|x_{\mathrm{avg}}^{\mathrm{c}}(k) - x_i^{\mathrm{c}}(k)\| + \|\nabla f_i(x_i^{\mathrm{c}}(k))\| \\
&\leq L\left(\|x_{\mathrm{avg}}^{\mathrm{c}}(k)\| + \|x_i^{\mathrm{c}}(k)\|\right) + \|\nabla f_i(x_i^{\mathrm{c}}(k))\| \\
&\leq 2L\mathcal{D} + G =: G^\star, \quad \forall k,
\end{aligned}$$

where we use $\|x_i^{\mathrm{c}}(k)\| \leq \|x^{\mathrm{c}}(k)\| \leq \mathcal{D}$ (as $x^{\mathrm{c}}(k)$ belongs to set $\mathcal{S}$), (4.6), and the Lipschitz continuity of the gradient ∇f_i (see Assumption 4.2.) Thus, the result in part (c). We now prove part (d). We have:

$$\begin{aligned}
\Psi_k^\star &= \sum_{i=1}^{N} f_i(x_i^{\mathrm{c}}(k)) + \frac{\rho}{2} k x^{\mathrm{c}}(k)^\top \mathcal{L} x^{\mathrm{c}}(k) \\
&\geq \sum_{i=1}^{N} \left(f_i(x_{\mathrm{avg}}^{\mathrm{c}}(k)) + \nabla f_i(x_{\mathrm{avg}}^{\mathrm{c}}(k))(x_i^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)) \right) \qquad (4.14) \\
&+ \frac{\rho}{2} k (x^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)\mathbf{1})^\top \mathcal{L} (x^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)\mathbf{1}) \\
&\geq f(x_{\mathrm{avg}}^{\mathrm{c}}(k)) + \sum_{i=1}^{N} \left(-G^\star \|x_i^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)\| + \frac{\rho}{2} k \lambda_2(\mathcal{L}) \|x_i^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)\|^2 \right) \qquad (4.15) \\
&\geq f^\star - \frac{N (G^\star)^2}{2\rho\, k\, \lambda_2(\mathcal{L})},
\end{aligned}$$

after (separately) minimizing each summand in (4.15) over $\epsilon := \|x_i^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)\| \in \mathbb{R}$. Inequality (4.14) used convexity of the f_i's and the fact that $\mathcal{L}(x_{\mathrm{avg}}^{\mathrm{c}}(k)\mathbf{1}) = 0$. Inequality (4.15) used the bound on the gradients $\|\nabla f_i(x_{\mathrm{avg}}^{\mathrm{c}}(k))\| \leq G^\star$, and the variational characterization of the eigenvalues to show $(x^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)\mathbf{1})^\top \mathcal{L}(x^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)\mathbf{1}) \geq \lambda_2(\mathcal{L})\|x^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)\mathbf{1}\|^2$, as $(x^{\mathrm{c}}(k) - x_{\mathrm{avg}}^{\mathrm{c}}(k)\mathbf{1})$ is orthogonal to

$q_1 = \frac{1}{\sqrt{N}}\mathbf{1}$–the eigenvector of $\mathcal{L}$ that corresponds to $\lambda_1(\mathcal{L}) = 0$. Thus, the result in part (d). $\square$

4.3.3 Convergence rate analysis

We briefly summarize the analysis. We first show that $\sum_{i=1}^N f_i(x_i(k)) = O(\log k)$, thus showing that $\sum_{i=1}^N f_i(x_i(k))$ does not grow fast with k. Then, using Assumption 4.3, we show that $\|\nabla f_i(y_i(k))\| = O(\log k)$. We then apply the latter to Theorem 2.8 with $G_k = O(\log k)$, that gives us that $f(x_i(k)) - f^\star = O(\log^3 k/k)$, and, as a corollary, that $f(x_i(k))$ is uniformly bounded, for all i, k. (Lemma 4.8.) Finally, we explain how to boost the optimality gap bound to $O(\log k/k)$.

Bounding the function values by $O(\log k)$

We now show that $\sum_{i=1}^N f_i(x_i(k))$ is $O(\log k)$.

Lemma 4.5 Consider algorithm (4.7)–(4.10) under Assumptions 4.1, 4.2 and 4.3. Further, denote by $R := \|\overline{x}(0) - x^\star\|$. Then, for all $k = 1, 2, ...$

$$\sum_{i=1}^N f_i(x_i(k)) \leq \frac{2\,N\,R^2}{c} + f(\overline{x}(0)) + 3\left(b\,N\,B - \sum_{i=1}^N f_i^\star\right) + \frac{N(G^\star)^2}{2\,\rho\,\lambda_2(\mathcal{L})} S_k,$$

where

$$S_k = 1 + \sum_{t=1}^{k-1} \frac{(t+1)^2}{t^3} + \sum_{t=2}^{k-1} \frac{1}{t\lfloor t/2 \rfloor} = O(\log k). \tag{4.16}$$

Proof: We prove Lemma 4.5 using the interpretation (4.12) that the iteration k of our algorithm (4.7)–(4.8) is a Nesterov gradient step with respect to Ψ_k. We use Lemma 2.5 from Chapter 2. We use it here to estimate the progress in one iteration with respect to Ψ_k. More precisely, denote by $v(k) = \frac{y(k) - (1-\theta_k)x(k)}{\theta_k}$ and $\theta_k = 2/(k+2)$. Applying Lemma 2.5 with $f \equiv \Psi_k$, $x^\bullet \equiv x^\star\mathbf{1}$, and $L_k = 1/\alpha_k = k/c$ (Note that here $\delta_k = 0$; also, we do not choose $x^\bullet$ to be an optimizer of Ψ_k, which is a valid choice):

$$\begin{aligned} \frac{(k+1)^2}{k}\left(\Psi_k(x(k)) - \Psi_k(x^\star\mathbf{1})\right) \quad &+ \quad (2/c)\|v(k) - x^\star\mathbf{1}\|^2 \\ &\leq \quad \frac{k^2-1}{k}\left(\Psi_k(x(k-1)) - \Psi_k(x^\star\mathbf{1})\right) + (2/c)\|v(k-1) - x^\star\mathbf{1}\|^2. \end{aligned} \tag{4.17}$$

Next, note that the term $\frac{k^2-1}{k}\left(\Psi_k(x(k-1)) - \Psi_k(x^\star \mathbf{1})\right)$ on the right hand side of (4.17) is, for $k = 1, 2, ...$, upper bounded as (because $\Psi_k(x^\star \mathbf{1}) \geq \Psi_k^\star$):

$$\begin{aligned} \frac{k^2-1}{k}\left(\Psi_k(x(k-1)) - \Psi_k(x^\star \mathbf{1})\right) &\leq \frac{k^2-1}{k}\left(\Psi_k(x(k-1)) - \Psi_k^\star\right) \\ &\leq k\left(\Psi_k(x(k-1)) - \Psi_k^\star\right), \end{aligned} \tag{4.18}$$

where the last inequality follows because $\Psi_k(x(k-1)) \geq \Psi_k^\star$. Further, the term $\frac{(k+1)^2}{k}\left(\Psi_k(x(k)) - \Psi_k(x^\star \mathbf{1})\right)$ on the left hand side of (4.17) is, for $k = 1, 2, ...$, lower bounded as:

$$\begin{aligned} \frac{(k+1)^2}{k}(\Psi_k(x(k)) - \Psi_k(x^\star \mathbf{1})) &= \frac{(k+1)^2}{k}\left(\Psi_k(x(k)) - \Psi_k^\star\right) - \frac{(k+1)^2}{k}\left(\Psi_k(x^\star \mathbf{1}) - \Psi_k^\star\right) \\ &\geq (k+2)\left(\Psi_k(x(k)) - \Psi_k^\star\right) - \frac{(k+1)^2}{k}\left(\Psi_k(x^\star \mathbf{1}) - \Psi_k^\star\right) \quad (4.19) \\ &\geq (k+2)\left(\Psi_k(x(k)) - \Psi_k^\star\right) - \frac{(k+1)^2}{k}\frac{N(G^\star)^2}{2\,\rho\,k\lambda_2(\mathcal{L})}, \quad (4.20) \end{aligned}$$

Here, (4.19) uses the fact that $\Psi_k(x(k)) - \Psi_k^\star \geq 0$, so that $\frac{k^2+2k+1}{k}(\Psi_k(x(k)) - \Psi_k^\star) \geq \frac{k^2+2k}{k}(\Psi_k(x(k)) - \Psi_k^\star)$; and (4.20) uses inequality $f^\star - \Psi_k^\star = \Psi_k(x^\star \mathbf{1}) - \Psi_k^\star \leq \frac{N(G^\star)^2}{2\rho k\lambda_2(\mathcal{L})}$ from Lemma 4.4, part (d). Using (4.18) and (4.20), dividing (4.17) by k, and rearranging the terms:

$$\begin{aligned} &\left(1 + \frac{2}{k}\right)\left(\Psi_k(x(k)) - \Psi_k^\star\right) + \frac{2}{ck}\|v(k) - x^\star \mathbf{1}\|^2 \\ \leq\ &\left(\Psi_k(x(k-1)) - \Psi_k^\star\right) + \frac{2}{ck}\|v(k-1) - x^\star \mathbf{1}\|^2 + \frac{(k+1)^2}{k^3}\frac{N(G^\star)^2}{2\,\rho\,\lambda_2(\mathcal{L})}, \end{aligned}$$

or, equivalently:

$$\begin{aligned} \left(\Psi_k(x(k)) - \Psi_k^\star\right) + \frac{2}{ck}\|v(k) - x^\star \mathbf{1}\|^2 &\leq \left(\Psi_k(x(k-1)) - \Psi_k^\star\right) + \frac{2}{ck}\|v(k-1) - x^\star \mathbf{1}\|^2 \\ &+ \frac{(k+1)^2}{k^3}\frac{N(G^\star)^2}{2\,\rho\,\lambda_2(\mathcal{L})} - \frac{2}{k}\left(\Psi_k(x(k)) - \Psi_k^\star\right). \end{aligned} \tag{4.21}$$

We next replace the term $\left(\Psi_k(x(k)) - \Psi_k^\star\right)$ on the left hand side in (4.21) with its lower bound that involves $\left(\Psi_{k+1}(x(k)) - \Psi_{k+1}^\star\right)$. Using the definition of the functions Ψ_k and Ψ_{k+1}, adding and subtracting $\Psi_{k+1}^\star +$

$\frac{\rho}{2}x(k)^\top \mathcal{L}x(k)$, and using the relation $\Psi^\star_{k+1} \geq \Psi^\star_k$ (see Lemma 4.4, part (b)):

$$\begin{aligned}
(\Psi_k(x(k)) - \Psi^\star_k) &= \sum_{i=1}^{N} f_i(x_i(k)) + \frac{\rho\, k}{2}x(k)^\top \mathcal{L}x(k) - \Psi^\star_k + \Psi^\star_{k+1} - \Psi^\star_{k+1} \\
&+ \frac{\rho}{2}x(k)^\top \mathcal{L}x(k) - \frac{\rho}{2}x(k)^\top \mathcal{L}x(k) \\
&= \left(\Psi_{k+1}(x(k)) - \Psi^\star_{k+1}\right) - \frac{\rho}{2}x(k)^\top \mathcal{L}x(k) + \left(\Psi^\star_{k+1} - \Psi^\star_k\right) \\
&\geq \left(\Psi_{k+1}(x(k)) - \Psi^\star_{k+1}\right) - \frac{\rho}{2}x(k)^\top \mathcal{L}x(k). \qquad (4.22)
\end{aligned}$$

Further, we replace the term $-\frac{2}{k}\ (\Psi_k(x(k)) - \Psi^\star_k)$ on the right hand side of (4.21) by an upper bound as follows. We express $(\Psi_k(x(k)) - \Psi^\star_k)$ in terms of $\left(\Psi_{\lfloor k/2\rfloor}(x(k)) - \Psi^\star_{\lfloor k/2\rfloor}\right)$ as follows:

$$\begin{aligned}
(\Psi_k(x(k)) - \Psi^\star_k) &= \sum_{i=1}^{N} f_i(x_i(k)) + \frac{\rho\, k}{2}x(k)^\top \mathcal{L}x(k) - \Psi^\star_k \\
&= \sum_{i=1}^{N} f_i(x_i(k)) + \frac{\rho\, \lfloor k/2\rfloor}{2}x(k)^\top \mathcal{L}x(k) - \Psi^\star_{\lfloor k/2\rfloor} + \Psi^\star_{\lfloor k/2\rfloor} - \Psi^\star_k + \frac{\rho(k - \lfloor k/2\rfloor)}{2}x(k)^\top \mathcal{L}x(k) \\
&= (\Psi_{\lfloor k/2\rfloor}(x(k)) - \Psi^\star_{\lfloor k/2\rfloor}) - (\Psi^\star_k - \Psi^\star_{\lfloor k/2\rfloor}) + \frac{\rho(k - \lfloor k/2\rfloor)}{2}x(k)^\top \mathcal{L}x(k).
\end{aligned}$$

Thus, using $(\Psi_{\lfloor k/2\rfloor}(x(k)) - \Psi^\star_{\lfloor k/2\rfloor}) \geq 0$, and $k - \lfloor k/2\rfloor \geq k/2$, the term $(\Psi_k(x(k)) - \Psi^\star_k)$ is bounded from above as:

$$(\Psi_k(x(k)) - \Psi^\star_k) \geq -(\Psi^\star_k - \Psi^\star_{\lfloor k/2\rfloor}) + \rho\frac{(k/2)}{2}x(k)^\top \mathcal{L}x(k),$$

or, equivalently:

$$-\frac{2}{k}\,(\Psi_k(x(k)) - \Psi^\star_k) \leq \frac{2}{k}(\Psi^\star_k - \Psi^\star_{\lfloor k/2\rfloor}) - \frac{\rho}{2}x(k)^\top \mathcal{L}x(k).$$

Next, by Lemma 4.4, parts (c) and (d), the term :

$$(\Psi^\star_k - \Psi^\star_{\lfloor k/2\rfloor}) = (\Psi^\star_k - f^\star) + (f^\star - \Psi^\star_{\lfloor k/2\rfloor}) \leq \frac{N(G^\star)^2}{2\,\rho\,\lambda_2(\mathcal{L})\lfloor k/2\rfloor},\ k = 2, 3, ...,$$

which finally gives:

$$-\frac{2}{k}\ (\Psi_k(x(k)) - \Psi^\star_k) \leq \left(\Psi^\star_k - \Psi^\star_{\lfloor k/2\rfloor}\right) - \frac{\rho}{2}x(k)^\top \mathcal{L}x(k) \leq \frac{N(G^\star)^2}{\rho\,\lambda_2(\mathcal{L})k\lfloor k/2\rfloor} - \frac{\rho}{2}x(k)^\top \mathcal{L}x(k),\ k = 2, 3, ... \qquad (4.23)$$

Note that, for $k = 1$:

$$-\frac{2}{k}\left(\Psi_k(x(k)) - \Psi_k^\star\right) \leq 2(\Psi_1^\star - \Psi_0^\star) - \frac{\rho}{2}x(1)^\top \mathcal{L} x(1). \tag{4.24}$$

Now, applying the bounds (4.22) and (4.23) to (4.21), for $k = 2, 3, ...$:

$$\begin{aligned}\left(\Psi_{k+1}(x(k)) - \Psi_{k+1}^\star\right) + \frac{2}{ck}\|v(k) - x^\star \mathbf{1}\|^2 &\leq \left(\Psi_k(x(k-1)) - \Psi_k^\star\right) + \frac{2}{ck}\|v(k-1) - x^\star \mathbf{1}\|^2 \\ &+ \frac{N(G^\star)^2}{2\lambda_2(\mathcal{L})}\left(\frac{(k+1)^2}{2\,k^3} + \frac{1}{k\lfloor k/2 \rfloor}\right),\end{aligned}$$

which gives:

$$\begin{aligned}\left(\Psi_{k+1}(x(k)) - \Psi_{k+1}^\star\right) &\leq \left(\Psi_k(x(k-1)) - \Psi_k^\star\right) + \frac{2}{ck}\|v(k-1) - x^\star \mathbf{1}\|^2 - \frac{2}{ck}\|v(k) - x^\star \mathbf{1}\|^2 \\ &+ \left[\frac{(k+1)^2}{2\,k^3} + \frac{1}{k\lfloor k/2 \rfloor}\right]\frac{N(G^\star)^2}{\rho\,\lambda_2(\mathcal{L})},\ k = 2, 3, ...\end{aligned} \tag{4.25}$$

Also, for $k = 1$:

$$\begin{aligned}\left(\Psi_2(x(1)) - \Psi_2^\star\right) &\leq \left(\Psi_1(x(0)) - \Psi_1^\star\right) + \frac{2}{c}\|v(0) - x^\star \mathbf{1}\|^2 - \frac{2}{c}\|v(1) - x^\star \mathbf{1}\|^2 \\ &+ \frac{(1+1)^2}{2 \cdot 1^3}\frac{N(G^\star)^2}{\rho\,\lambda_2(\mathcal{L})} + 2(\Psi_1^\star - \Psi_0^\star).\end{aligned} \tag{4.26}$$

Finally, by telescoping (4.25) and (4.26), and using the definition of S_{k+1} in (4.16):

$$\begin{aligned}\left(\Psi_{k+1}(x(k)) - \Psi_{k+1}^\star\right) &\leq \Psi_1(x(0)) - \Psi_1^\star + (2/c)\|v(0) - x^\star \mathbf{1}\|^2 + \frac{N(G^\star)^2}{\rho\,\lambda_2(\mathcal{L})}\left[S_{k+1}\right] \\ &+ 2(\Psi_1^\star - \Psi_0^\star).\end{aligned} \tag{4.27}$$

Use equality $x(0) = v(0) = \overline{x}(0)\mathbf{1}$; $\Psi_1^\star \leq f^\star$; $\Psi_0^\star \geq \sum_{i=1}^N f_i^\star$; $\Psi_1^\star \geq \sum_{i=1}^N f_i^\star$; and $\Psi_1(\overline{x}(0)\mathbf{1}) = f(\overline{x}(0))$. Substituting the latter findings in (4.27), we get the desired result. □

Bounding gradients by $O(\log k)$

We now use Lemma 4.5 to show that the gradients $\|\nabla f_i(y_i(k))\| = O(\log K)$, $k = 1, ..., K$. Denote by:

$$C_f := \left(f^\star - \min_{i=1,...,N}\sum_{j\neq i} f_j^\star\right) + 3\left(b\,N\,B - \sum_{i=1}^N f_i^\star\right) + f(\overline{x}(0)) + \frac{2\,N\,R^2}{c} + \frac{N(G^\star)^2}{2\,\rho\,\lambda_2(\mathcal{L})}. \tag{4.28}$$

Lemma 4.6 Consider algorithm (4.7)–(4.10) under Assumptions 4.1, 4.2 and 4.3, and denote by:

$$C_{\text{grad}} := 3L \max\left\{B, \frac{1}{b}C_f\right\} + L\|\overline{x}(0)\| + \max_{i=1,...,N} \|\nabla f_i(y_i(\overline{x}(0)))\|. \tag{4.29}$$

Then, for all $k = 0, 1, ..., K$:

$$\|\nabla f_i(y_i(k))\| \leq C_{\text{grad}}\, S_K.$$

Proof: Fix arbitrary node $i \in \{1, 2, ..., N\}$. By Lemma 4.5, and using $f_j(x_i(k)) \geq f_j^\star$, $j \neq i$:

$$\begin{aligned} f_i(x_i(k)) &\leq \left(f^\star - \sum_{j\neq i}^{N} f_j^\star\right) + \frac{2NR^2}{c} + \frac{N(G^\star)^2}{2\,\rho\,\lambda_2(\mathcal{L})} S_k + 3\left(bNB - \sum_{i=1}^{N} f_i^\star\right) \\ &\leq C_f\, S_k \leq C_f\, S_K, \end{aligned}$$

because $S_k \geq 1$ for all $k = 1, ..., K$, and $S_k \leq S_K$, for all $k = 1, ..., K$. Next, using Assumption 4.3:

$$\|x_i(k)\| \leq \max\{B, (1/b)C_f\}\ S_K.$$

which, because $\|y(k)\| = \|x(k) + \frac{k-1}{k+2}(x(k) - x(k-1))\| \leq 2\|x(k)\| + \|x(k-1)\|$, gives:

$$\|y_i(k)\| \leq 3\ \max\{B, (1/b)C_f\}\ S_K, \tag{4.30}$$

Now, using the Lipschitz continuity of ∇f_i and the triangle inequality:

$$\begin{aligned} \|\nabla f_i(y_i(k))\| &= \|\nabla f_i(y_i(k)) - \nabla f_i(y_i(0)) + \nabla f_i(y_i(0))\| \\ &\leq \|\nabla f_i(y_i(k)) - \nabla f_i(y_i(0))\| + \|\nabla f_i(y_i(0))\| \\ &\leq L\|y_i(k) - y_i(0)\| + \|\nabla f_i(y_i(0))\| \\ &\leq L\|y_i(k)\| + L\|y_i(0)\| + \|\nabla f_i(y_i(0))\|. \end{aligned}$$

The latter gives the desired result using the bound (4.66), the inequalities $\|y_i(0)\| = \|x_i(0)\| = \|\overline{x}(0)\|$, $\|\nabla f_i(y_i(0))\| \leq \max_{i=1,...,N} \|\nabla f_i(\overline{x}(0))\|$, and using $S_K \geq 1$. □

Optimality gap $O(\log^3 k/k)$: Bounding the function values by $O(1)$

We are now ready to prove the $O(\log^3 k/k)$ rate of convergence, as well as the bounded gradients result.

Theorem 4.7 (The $O(\log^3 k/k)$ rate of convergence under the growth assumption) Consider algorithm (4.7)–(4.10) under Assumptions 4.1, 4.2 and 4.3. Then, for all nodes i, the optimality gap $\frac{1}{N}(\, f(x_i(k)) \leq f^\star \,) = O(\log^3 k/k)$; more precisely:

$$\begin{aligned} & \frac{1}{N}\left(f(x_i(k)) - f^\star\right) \\ \leq \;\; & \frac{2\,R^2}{c} + 16\,LC_{\rm cons}^2 C_{\rm grad}^2\,\frac{S_k^2}{k}\,\sum_{t=1}^{k}\frac{(t+1)^2}{t^3} + C_{\rm grad}^2\,C_{\rm cons}\,\frac{S_k^2}{k},\; k=1,2,..., \end{aligned} \tag{4.31}$$

where S_k is in (4.16); $C_{\rm cons}$ is in (2.21); and $C_{\rm grad}$ is in (4.29).

Lemma 4.8 Consider algorithm (4.7)–(4.10) under Assumptions 4.1, 4.2 and 4.3. Then, for all nodes i, for all $k = 1, 2, ...$:

$$f(x_i(k)) \leq f^\star + \frac{2NR^2}{c} + a_1\,LC_{\rm cons}^2 C_{\rm grad}^2 + a_2\,NC_{\rm grad}^2 C_{\rm cons},$$

where S_k is in (4.16); $C_{\rm cons}$ is in (2.21); $C_{\rm grad}$ is in (4.29); and a_1, a_2 are universal constants independent of system parameters.

Proof: [Proof of Theorem 4.7] We recall Theorem 2.8 from Chapter 2. Recall that, to establish the optimality gap at iteration k, the proof of this Theorem actually required only that the gradients $\|\nabla f_i(y_i(t))\|$ be bounded, $\forall t = 0, 1, ..., k$. Hence, for a fixed k, we can replace the uniform bound on the gradients G with a bound G_k that satisfies: $\|\nabla f_i(y_i(t))\| \leq G_k$, $\forall t = 0, 1, ..., k$. We can use $G_k = C_{\rm grad}\,S_k$, with S_k in (4.16) and $C_{\rm grad}$ in (4.29). Applying Theorem 2.8 with $G_k = C_{\rm cons}\,S_k$, we get (4.31), and thus, Theorem 4.7. □

Proof: [Proof of Lemma 4.8] Lemma 4.8 follows after maximizing the right hand side in (4.31) over $k \geq 1$, i.e., after calculating that:

$$16\,\max_{k\geq 1}\left\{\frac{S_k^2}{k}\sum_{t=1}^{k}\frac{(t+1)^2}{t^3}\right\} \leq 2000, \quad \max_{k\geq 1}\left\{\frac{S_k^2}{k}\right\} \leq 50. \quad \square$$

Improving convergence rate to $O(\log k/k)$

It is clear that we can now improve convergence rate to $O(\log k/k)$. As the function values $f(x_i(k))$ are uniformly bounded by a constant for all k, we proceed like in the proof of Lemma 4.6, and conclude that the gradients $\nabla f_i(y_i(k))$ are uniformly bounded by a constant, i.e., it holds that: $\|\nabla f_i(y_i(k))\| \leq C'_{\rm grad}, \forall i, \forall k,$

for a certain constant C'_{grad}. Applying Theorem 2.8 from Chapter 2, we obtain the $O(\log k/k)$ convergence rate, as desired.

4.4 D–NG method: Constrained optimization with constant step-size

In this Section, we present a projected D–NG method for constrained optimization (4.1). Subsection 4.4.1 introduces the model, the algorithm, and clone functions similar to the previous Section; Subsection 4.4.2 performs convergence analysis. This Section assumes static networks.

4.4.1 Model and algorithm

We impose the following structure on the costs f_i's and the constraint set $\mathcal{X}$:

Assumption 4.9 (a) For all i, f_i is convex, coercive, and Lipschitz with respect to the Euclidean $\|\cdot\|$ norm on the set $\mathcal{X}$, i.e., there exists $G' \in (0, \infty)$, such that:

$$\|f_i(x) - f_i(y)\| \leq G' \, \|x - y\| \, , \, \forall x, y \in \mathcal{X}.$$

(b) f_i is continuously differentiable, with Lipschitz continuous first derivative of constant L:

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L \, \|x - y\|, \quad \forall x, y \in \mathbb{R}^d.$$

(c) The set $\mathcal{X}$ is closed, convex, and non-empty.

Condition 1 (a) on the Lipschitz continuity of $f_i(\cdot)$ on $\mathcal{X}$ holds for any function $f_i(\cdot)$ that satisfies the other Assumptions in 1 when $\mathcal{X}$ is a compact set. By Assumption 4.9, problem (3.5) is solvable, the optimal value $f^\star > -\infty$, and the solution set is non-empty and compact, e.g., [81].

The algorithm

We now present the projected D–NG algorithm. The algorithm is the same as (4.7)–(4.8), except that it introduces the projection step:

$$x_i(k) = P_{\mathcal{X}} \left\{ (1 - \ell_i \, w) \, y_i(k-1) + w \sum_{j \in O_i} y_j(k-1) - \alpha \nabla f_i(y_i(k-1)) \right\} \tag{4.32}$$

$$y_i(k) = x_i(k) + \beta_{k-1} \, (x_i(k) - x_i(k-1)) \, , \; k = 1, 2, ..., \tag{4.33}$$

with initialization $x_i(0) = y_i(0) = \overline{x}(0)$. Here, β_k is given in (4.9), and $P_{\mathcal{X}}$ is the Euclidean projection onto the set $\mathcal{X}$:

$$P_{\mathcal{X}}\{z_i\} = \arg\min_{y_i \in \mathcal{X}} \|y_i - z_i\|^2.$$

The variables $x_i(k)$ are feasible, i.e., $x_i(k) \in \mathcal{X}$, for all k, while the variables $y_i(k)$ may not be feasible.

We require that the step size α and the averaging weight w satisfy:

$$\alpha \leq \frac{1}{L + \rho\lambda_N(\mathcal{L})} \tag{4.34}$$

$$w = \alpha\,\rho, \tag{4.35}$$

where $\rho > 0$ is a parameter specified further ahead. Hence, we require the same a priori knowledge (parameters L and $\lambda_N(\mathcal{L})$) as in Section 4.3.

Clone functions Ψ_ρ

Similarly to Section 4.3, we introduce the clone function Ψ_ρ and the clone problem as:

$$\begin{array}{ll} \text{minimize} & \Psi_\rho(x) := \sum_{i=1}^N f_i(x_i) + \frac{\rho}{2}\, x^\top \mathcal{L} x \\ \text{subject to} & x \in \mathcal{X}^N, \end{array} . \tag{4.36}$$

where $\mathcal{X}^N$ denotes the Cartesian product $\mathcal{X}^N = \mathcal{X} \times ... \times \mathcal{X}$ ($\mathcal{X}$ repeated N times.) By Assumption 4.9, problem (4.11) is solvable and has a compact solution set. Denote by $\Psi_\rho^\star$ the optimal value of (4.11), and $x^{\mathrm{c}}(\rho)$ a solution to (4.11). By Lemma 4.4, we have that $\Psi_\rho^\star \leq f^\star$.

4.4.2 Convergence analysis

We now study convergence of the projected D–NG algorithm (4.32)–(4.33). We have the following Theorem.

Theorem 4.10 Consider algorithm (4.32)–(4.34) under Assumptions 4.1 and 4.9, with the step-size

$$\alpha = \frac{1}{L + \rho\lambda_N(\mathcal{L})}.$$

Then:

(a) for all $i = 1, ..., N$, for all $k = 1, 2, ...$:

$$frac1N\,(\,f(x_i(k)) - f^\star\,) \leq \frac{N(N-1)(G')^2}{2\,\rho} + \frac{[2\;(L + \rho\lambda_N(\mathcal{L}))]\;\|x(0) - x^{\rm c}(\rho)\|^2}{N\,k^2}. \qquad (4.37)$$

(b) Let $\mathcal{X}$ be a compact set with $\|y\| \leq B'$, for all $y \in \mathcal{X}$, and fix the desired accuracy $\epsilon > 0$. Then, for:

$$\rho = \rho(\epsilon) = \frac{N(G')^2(N-1)}{\epsilon},$$

we have: $\frac{1}{N}(\,f(x_i(k)) - f^\star\,) \leq \epsilon, \forall k \geq k_0(\epsilon), \forall i$, where:

$$k_0(\epsilon) = \frac{4N\sqrt{\lambda_N(\mathcal{L})}G'B'}{\epsilon} + \frac{4B'\sqrt{\max_i L}}{\sqrt{\epsilon}},$$

i.e., the ϵ-accuracy is achieved after at most $k_0(\epsilon)$ iterations.

Theorem 4.10 says that, with the proposed D–NG algorithm, for the compact set $\mathcal{X}$ and appropriately chosen ρ, the number of iterations (per-node communications and per-node gradient evaluations) for ϵ-accuracy in the cost function is $O(1/\epsilon)$.

Proof: [Proof of Theorem 4.10] We first prove claim (a). The proof consists of two parts. First, we use the convergence results for the Nesterov method [73, 82] to estimate the error in terms of the clone function $\Psi_\rho(x(k)) - \Psi_\rho^\star$. Second, we relate the clone error $\Psi_\rho(x(k)) - \Psi_\rho^\star$ and the true error at any node j: $f(x_j(k)) - f^\star$.

Clone function error

By the convergence results for the Nesterov gradient method [73], and noting that the Lipschitz constant of Ψ_ρ equals $L + \rho\lambda_N(\mathcal{L})$, we have that, for all k:

$$\Psi_\rho(x(k)) - \Psi_\rho^\star \leq \frac{[2\;(L + \rho\lambda_N(\mathcal{L}))]\,\|x(0) - x^{\rm c}(\rho)\|^2}{k^2} =: \frac{C_\Psi}{k^2}.$$

Relating the clone and the true errors

We now fix a node j and start with the clone error:

$$\Psi_\rho(x(k)) - \Psi_\rho^\star \qquad (4.38)$$

$$= \sum_{i=1}^{N} f_i(x_i(k)) + \frac{1}{2}\,\rho\,x(k)^\top\mathcal{L}\,x(k) - \Psi_\rho^\star. \qquad (4.39)$$

Consider equation (4.38), and fix a node j at which we want to estimate the true error. By Lipschitz continuity of $f_i(\cdot)$ on $\mathcal{X}$ and by the fact that $x_i(k), x_j(k) \in \mathcal{X}$, we have that:

$$\|f_i(x_i(k)) - f_i(x_j(k))\| \leq G'\|x_i(k) - x_j(k)\|. \tag{4.40}$$

Now, adding and subtracting $f^\star$ from (4.38) while using the fact that $\Psi_\rho^\star \leq f^\star$, and using (4.40) gives:

$$\begin{aligned}
& \Psi_\rho(x(k)) - \Psi_\rho^\star \geq \sum_{i=1}^{N} f_i(x_j(k)) - f^\star \qquad (4.41)\\
- \quad & \sum_{i=1}^{N} G'\|x_i(k) - x_j(k)\| + \frac{1}{2}\,\rho\, x(k)^\top \mathcal{L} x(k)\\
\geq \quad & f(x_j(k)) - f^\star - N\,G'\left(\max_{i:\, i\neq j}\|x_i(k) - x_j(k)\|\right)\\
+ \quad & \frac{1}{2}\,\rho\, x(k)^\top \mathcal{L} x(k).
\end{aligned}$$

We now lower bound the quadratic form

$$x^\top \mathcal{L} x = \sum_{\{i,j\}\in E} \|x_i - x_j\|^2,$$

for any $x \in \mathbb{R}^N$. Fix a node j, and let $\max_{i:\, i\neq j}\|x_i - x_j\| =: \|x_s - x_j\|$. Because the graph is connected, there is a path of length D from node s to node j, say $(s = i_1) \to i_2 \to \ldots \to (i_{D+1} = j)$, where $1 \leq D \leq N-1$. Then:

$$\begin{aligned}
& x^\top \mathcal{L} x \geq \|x_s - x_{i_2}\|^2 + \ldots + \|x_{i_D} - x_j\|^2\\
= \quad & D\left(\frac{1}{D}\|x_s - x_{i_2}\|^2 + \ldots + \frac{1}{D}\|x_{i_D} - x_j\|^2\right)\\
\geq \quad & D\left\|\frac{1}{D}(x_s - x_{i_2}) + \ldots + \frac{1}{D}(x_{i_D} - x_j)\right\|^2 \qquad (4.42)\\
= \quad & \frac{1}{D}\|x_s - x_j\|^2 \geq \frac{1}{(N-1)}\|x_s - x_j\|^2,
\end{aligned}$$

where we use the fact that, for any path, $D \leq (N-1)$, and inequality (4.42) uses convexity of the quadratic function $z \mapsto \|z\|^2$. Using the latter bound for $x = x(k)$, we have:

$$\begin{aligned} & \Psi_\rho(x(k)) - \Psi_\rho^\star \geq f(x_j(k)) - f^\star \\ - \quad & NG' \left(\max_{i:i\neq j} \|x_i(k) - x_j(k)\| \right) \\ + \quad & \frac{1}{2} \frac{\rho}{(N-1)} \left(\max_{i:i\neq j} \|x_i(k) - x_j(k)\| \right)^2 \qquad (4.43) \\ \geq \quad & f(x_j(k)) - f^\star - \frac{N^2(N-1)(G')^2}{2\,\rho}, \qquad (4.44) \end{aligned}$$

where (4.44) follows by maximizing $NG'\delta - \frac{1}{2}\frac{\rho}{(N-1)}\delta^2$ over $\delta \in \mathbb{R}$. Equation (4.44) allows us to relate the clone and the true errors:

$$f(x_j(k)) - f^\star \leq \Psi_k(x(k)) - \Psi_\rho^\star + \frac{N^2(N-1)(G')^2}{2\,\rho}. \qquad (4.45)$$

Equation (4.45), combined with (4.38), and dividing both sides of the inequality by N, completes the proof of part (a).

We now prove part (b). Let the set $\mathcal{X}$ be compact, such that $\|y\| \leq B'$, for all $y \in \mathcal{X}$. Denote by $B := \sqrt{N}B'$. Then,

$$\|x(0) - x^{\mathrm{c}}(\rho)\| \leq \|x(0)\| + \|x^{\mathrm{c}}(\rho)\| \leq 2B,$$

which gives:

$$\begin{aligned} & f(x_i(k)) - f^\star \leq \frac{N^2(N-1)(G')^2}{2\,\rho} \qquad (4.46) \\ + \quad & \frac{[2\ (L + \rho\lambda_N(\mathcal{L}))]\ (2B)^2}{k^2} \\ = \quad & \frac{C_1}{\rho} + \rho\frac{C_2}{k^2} + \frac{C_3}{k^2}, \qquad (4.47) \end{aligned}$$

with $C_1 = \frac{N^2(G')^2(N-1)}{2}$, $C_2 = 8B^2\lambda_N(\mathcal{L})$, and $C_3 = 8LB^2$. Now, fix an $\epsilon > 0$, and consider the iteration $K(\rho)$–the smallest iteration k at which $f(x_i(k)) - f^\star \leq \epsilon$, for all i. Our goal is then to find $\rho > 0$ that minimizes $K(\rho)$ and to find the corresponding minimal value $K^\star(\epsilon)$. Instead of finding the actual minimum, it suffices for our purpose to find an upper bound on $K^\star(\epsilon)$, and a sub-optimal ρ, which we call

$\rho(\epsilon)$. By (4.47), we have that

$$K^2(\rho) \leq \frac{\rho^2 C_2 + \rho\, C_3}{\epsilon\rho - C_1} =: \mathcal{M}(\epsilon, \rho),\ \rho > C_1/\epsilon. \tag{4.48}$$

Now, set $\rho(\epsilon) := \frac{2C_1}{\epsilon} = \frac{N^2(G')^2(N-1)}{\epsilon}$. This value, when plugged in the right hand side of (4.48), gives:

$$\mathcal{M}(\epsilon, \rho(\epsilon)) = \frac{4C_1C_2}{\epsilon^2} + \frac{2C_3}{\epsilon}.$$

From above, using inequality $\sqrt{x+y} \leq \sqrt{x} + \sqrt{y}$, $x, y \geq 0$, we can conclude that:

$$K^\star(\epsilon) \leq \sqrt{\mathcal{M}(\epsilon, \rho(\epsilon))} \leq \frac{2\sqrt{C_1C_2}}{\epsilon} + \frac{\sqrt{2C_3}}{\sqrt{\epsilon}}.$$

Substituting the values of C_1, C_2, and C_3, and replacing the value of ϵ to $N\,\epsilon$ (because we are interested in the normalized optimality gap $\frac{1}{N}(\, f(x_i) - f^\star\,)$), we obtain the result (b). □

Remark. For the projected D–NG method with a *constant step size*, the clone function approach gives better bounds than the approach using "global averages" $\overline{x}(k)$ and $\overline{y}(k)$ that we adopted in Chapters 2 and 3. Namely, with the "global averages" approach, we would get (we do not consider here the constants, expect the step size α, to illsustrate the point):

$$f(\overline{x}(k)) - f^\star = O\left(\frac{1}{k^2}\right) + \alpha\, O\left(\frac{1}{k^2}\sum_{t=1}^{k} t^2 \|\widetilde{y}(t)\|\right),$$

where $\overline{x}(k) = \frac{1}{N}\sum_{i=1}^N x_i(k)$, $\overline{y}(k) = \frac{1}{N}\sum_{i=1}^N y_i(k)$, and $\widetilde{y}(k) = y(k) - \overline{y}(k)\mathbf{1}$. As the disagreement is not guaranteed to converge to zero under a constant step-size, the second summand above is $O(\alpha\, k)$, and we get the error accumulation. This, however, does not reflect the actual behavior of the projected D–NG method, whereby the cost function stays in a neighborhood of $f^\star$ when k increases.

4.5 Projected mD–NC method: Constrained optimization

This Section presents the projected mD–NC method for random networks. In Subsection 4.5.1, we introduce the framework of (centralized) inexact projected Nesterov gradient. This extends our results from Chapter 2 to *constrained* problems and inexact, feasible projections. Subsection 4.5.2 introduces the model and presents the projected mD–NC method. Finally, Subsection 4.5.3 performs convergence rate analysis.

4.5.1 Framework of Inexact Nesterov gradient method

Throughout this Subsection, we consider the (centralized) constrained minimization of a function $f(x)$ subject to $x \in \mathcal{X}$, where $f : \mathbb{R}^d \to \mathbb{R}$ is convex, and $\mathcal{X} \subset \mathbb{R}^d$ is a nonempty, closed, convex set. Before detailing the inexact projected Nesterov gradient, we recall from Chapter 2 the pointwise inexact (first order) oracle and the inexact projection.

Definition 4.11 (Inexact oracle) Consider a convex function $f : \mathbb{R}^d \to \mathbb{R}$ and a nonempty, closed, convex set $\mathcal{X}$. We say that a pair $\left(\widehat{f}_y, \widehat{g}_y\right) \in \mathbb{R} \times \mathbb{R}^d$ is a (L_y, δ_y) inexact oracle of f at point $y \in \mathbb{R}^d$ over the set $\mathcal{X}$ if:

$$f(x) \geq \widehat{f}_y + \widehat{g}_y^\top (x - y), \quad \forall x \in \mathcal{X} \tag{4.49}$$

$$f(x) \leq \widehat{f}_y + \widehat{g}_y^\top (x - y) + \frac{L_y}{2}\|x - y\|^2 + \delta_y, \quad \forall x \in \mathcal{X}. \tag{4.50}$$

We give a couple of remarks with respect to Definition 4.11. First, in Definition 4.11, we require that x belongs to $\mathcal{X}$, while y may lie outside $\mathcal{X}$. Second, throughout we just use the wording "inexact oracle at y" rather than "inexact oracle of f at y over $\mathcal{X}$," as the set $\mathcal{X}$ and the function f are understood from context. Finally, if $\left(\widehat{f}_y, \widehat{g}_y\right)$ is a (L_y, δ_y) inexact oracle at y, then it is also a $\left(L'_y, \delta_y\right)$ inexact oracle at y, with $L'_y \geq L_y$.

We next give the definition of an inexact projection. First, denote the exact (Euclidean) projection of $y \in \mathbb{R}^d$ on $\mathcal{X}$ by $P_{\mathcal{X}}\{y\} = \arg\min_{z \in \mathcal{X}} \|z - y\|$.

Definition 4.12 (Inexact projection) We say that $x \in \mathbb{R}^d$ is a ζ-inexact projection of $y \in \mathbb{R}^d$ on $\mathcal{X}$ if: 1) $x \in \mathcal{X}$; and 2) $\|x - P_{\mathcal{X}}\{y\}\| \leq \zeta$.

Inexact projected Nesterov gradient

We consider the following inexact iteration of the Nesterov gradient method to minimize $f(x)$ over $\mathcal{X}$. For a given point $(\overline{x}(k-1), \overline{y}(k-1)) \in \mathcal{X} \times \mathbb{R}^d$, let $\left(\widehat{f}_{k-1}, \widehat{g}_{k-1}\right)$ be a (L_{k-1}, δ_{k-1}) inexact oracle at $\overline{y}(k-1)$; further, let $\widehat{\mathcal{P}}_k \left\{\overline{y}(k-1) - \frac{1}{L_{k-1}}\widehat{g}_{k-1}\right\}$ be a ζ_{k-1}-inexact projection of $\overline{y}(k-1) - \frac{1}{L_{k-1}}\widehat{g}_{k-1}$. Construct $\overline{x}(k), \overline{y}(k)$ as:

$$\overline{x}(k) = \widehat{\mathcal{P}}_k \left\{\overline{y}(k-1) - \frac{1}{L_{k-1}}\widehat{g}_{k-1}\right\} \tag{4.51}$$

$$\overline{y}(k) = \overline{x}(k) + \beta_{k-1}\left(\overline{x}(k) - \overline{x}(k-1)\right). \tag{4.52}$$

We have the following Lemma on the progress (per iteration) of (4.51)–(4.52).

Lemma 4.13 (Progress per iteration) Consider (4.51)–(4.52) for some $k = 1, 2, ...$ and let $x^\bullet$ be arbitrary point in $\mathcal{X}$. Further, assume that the set $\mathcal{X}$ is compact with $\|x\| \leq B, \forall x \in \mathcal{X}$, Then:

$$\begin{aligned} & (k+1)^2 \ (f(\overline{x}(k)) - f(x^\bullet)) + 2L_{k-1}\|\overline{v}(k) - x^\bullet\|^2 \qquad (4.53) \\ \leq \ & (k^2-1) \ (f(\overline{x}(k-1)) - f(x^\bullet)) + 2L_{k-1}\|\overline{v}(k-1) - x^\bullet\|^2 + (k+1)^2\delta_{k-1} + (k+1)^2\eta_{k-1}, \end{aligned}$$

where $\theta_k = 2/(k+2)$ and

$$\overline{v}(k-1) = \frac{\overline{y}(k-1) - (1-\theta_{k-1})\overline{x}(k-1)}{\theta_{k-1}} \qquad (4.54)$$

$$\eta_{k-1} = L_{k-1}\zeta_{k-1}^2 + L_{k-1}\left(6B + \frac{\|\widehat{g}_{k-1}\|}{L_{k-1}}\right)\zeta_{k-1} \qquad (4.55)$$

Proof of Lemma 4.13 is in Section 4.6.

4.5.2 Model and algorithm

We consider constrained optimization problem (4.1) and let the f_i's and $\mathcal{X}$ obey the following.

Assumption 4.14 (a) The set $\mathcal{X}$ is nonempty, convex, and compact with $\|x\| \leq B$, $\forall x \in \mathcal{X}$ for some $B \in (0, \infty)$.

(b) For all i, $f_i : \mathbb{R}^d \mapsto \mathbb{R}$ is convex, continuously differentiable, with Lipschitz continuous gradient with constant L on the set $\mathcal{X}' := \left\{x \in \mathbb{R}^d : \|x\| \leq 3\,B\right\}$:

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|, \ \forall x, y \in \mathcal{X}'.$$

By Assumption 4.14, there exists a solution $x^\star \in \mathcal{X}$ with $f(x^\star) = f^\star = \inf_{x\in\mathcal{X}} f(x)$, and the solution set $\{x^\star \in \mathcal{X} : \ f(x^\star) - f^\star\}$ is compact. Also, the gradient $\nabla f_i(x)$ is bounded over the set $\mathcal{X}'$, i.e., there exists a constant $G \in [0, \infty)$ such that $\|\nabla f_i(x)\| \leq G, \forall x \in \mathcal{X}'$. Assumption 4.14 encompasses many costs f_i's; e.g., any f_i that is twice continuously differentiable on $\mathbb{R}^d$ obeys Assumption 4.14 (b) with constant $L = \max_{x\in\mathcal{X}'} \|\nabla^2 f_i(x)\|$.

For convenience, we briefly recall the random network model in Chapter 2. The projected mD–NC algorithms operates in two time scales, i.e., it has inner iterations s and outer iterations k. There are τ_k inner iterations at the outer iteration s, with τ_k specified further ahead. We capture the communication pattern at (k, s) by the random matrix $W(k, s)$ that obeys the following.

Assumption 4.15 The matrices $W(k,s)$ are:

(a) Mutually independent and identically distributed;

(b) Stochastic, symmetric, with positive diagonals, almost surely;

(c) There exists a positive number $\underline{w}$, such that, for all $i,j = 1,...,N$, almost surely, $W_{ij}(k,s) \in \{0\} \cup [\underline{w}, 1]$;

(d) The graph is connected on average, i.e., $\|\mathbb{E}\left[W(k,s)\right] - J\| < 1$.

Denote by $\overline{\mu} := \left(\|\mathbb{E}\left[W(k,s)^2\right] - J\|\right)^{1/2}$, and introduce, for future reference, the following matrices:

$$\mathcal{W}(k) = W(k,\tau_k)W(k,\tau_k-1)...W(k,1) \;\text{ and }\; \widetilde{\mathcal{W}}(k) := \mathcal{W}(k) - J. \tag{4.56}$$

Projected mD–NC algorithm

Projected mD–NC has all equal steps as the algorithm mD–NC in Chapter 2, and an additional projection on the constraint set $\mathcal{X}$. (Recall that $P_{\mathcal{X}}\{y\}$ denotes the projection of y on $\mathcal{X}$.) The projected mD–NC is summarized in Algorithm 3. The step-size $\alpha \leq 1/(2L)$.

Algorithm 3 The projected mD–NC algorithm

1: Initialization: Node i sets: $x_i(0) = y_i(0) \in \mathbb{R}^d$; and $k = 1$.
2: Node i calculates: $x_i^{(a)}(k) = y_i(k-1) - \alpha \nabla f_i(y_i(k-1))$.
3: (Consensus) Nodes run average consensus on a $2\,d \times 1$ variable $\chi_i(s,k)$, initialized by $\chi_i(s=0,k) = (x_i^{(a)}(k)^\top, x_i(k-1)^\top)^\top$:

$$\chi_i(s,k) = \sum_{j \in O_i(k)} W_{ij}(k,s)\,\chi_j(s-1,k), \;\; s = 1,2,...,\tau_k, \qquad \tau_k = \left\lceil \frac{4\log k + \log N}{-\log \bar{\mu}} \right\rceil, \tag{4.57}$$

and set $x_i^{(c)}(k) := [\chi_i(s=\tau_k,k)]_{1:d}$ and $x_i^{(b)}(k-1) := [\chi_i(s=\tau_k,k)]_{d+1:2\,d}$. (Here $[a]_{l:m}$ is a selection of l-th, $l+1$-th, ..., m-th entries of vector a.)
4: Node i calculates:

$$x_i(k) := P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\}.$$

5: Node i calculates:

$$y_i(k) = (1+\beta_{k-1})x_i(k) - \beta_{k-1}\,x_i^{(b)}(k-1).$$

6: Set $k \mapsto k+1$ and go to step 2.

4.5.3 Convergence rate analysis

Inexact oracle framework

To analyze the convergence rate of the projected mD–NC algorithm, we use the framework of inexact projected Nesterov gradient method in Subsection 2.4.1. Similarly to mD–NC for the unconstrained optimization in Chapter 2, we consider the global average $\overline{x}(k) := \frac{1}{N}\sum_{i=1}^{N} x_i(k)$, the disagreement at node i: $\widetilde{x}_i(k) := x_i(k) - \overline{x}(k)$, and the aggregate quantities $x(k) := (x_1(k)^\top, ..., x_N(k)^\top)^\top$ and $\widetilde{x}(k) := (x_1(k)^\top, ..., x_N(k)^\top)^\top$. We also consider the counterparts for $y_i(k)$, $x_i^{(a)}(k)$, $x_i^{(b)}(k)$, and $x_i^{(c)}(k)$, defined analogously.

We next derive the update equation for $(\overline{x}(k), \overline{y}(k))$. From Algorithm 3, steps 2 and 3, we have that $\overline{x}^{(a)}(k) = \overline{x}^{(c)}(k) = \overline{y}(k) - \frac{\alpha}{N}\sum_{i=1}^{N} \nabla f_i(y_i(k-1))$; from the latter and steps 4 and 5:

$$\overline{x}(k) = \widehat{\mathcal{P}}_k \left\{ \overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N} \nabla f_i(y_i(k-1)) \right\} \tag{4.58}$$

$$\overline{y}(k) = \overline{x}(k) + \beta_{k-1}\left(\overline{x}(k) - \overline{x}(k-1)\right), \tag{4.59}$$

where we define the inexact projection $\widehat{\mathcal{P}}_k$ by:

$$\widehat{\mathcal{P}}_k \left\{ \overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N} \nabla f_i(y_i(k-1)) \right\} = \widehat{\mathcal{P}}_k \left\{ \overline{x}^{(c)}(k) \right\} := \frac{1}{N}\sum_{i=1}^{N} P_{\mathcal{X}} \left\{ x_i^{(c)}(k) \right\}. \tag{4.60}$$

As with mD–NC for unconstrained optimization, algorithm (4.58)–(4.59) can be viewed as an inexact projected Nesterov gradient algorithm. Both the "gradient direction" and the projection step are inexact. With respect to "gradient direction" inexactness, we recall Lemma 2.5 in Chapter 2. This Lemma continues to hold here as well. Furthermore, it can be shown that we still have $\delta_{k-1} := L\|\widetilde{y}(k-1)\|^2$. The next Lemma quantifies the projection inexactness.

Lemma 4.16 Consider the projected mD–NC algorithm with step size $\alpha \leq 1/(2L)$, and let Assumptions 4.14 and 4.15 hold. Then, $\overline{x}(k)$ is ζ_{k-1}–inexact projection of $\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N} \nabla f_i(y_i(k-1))$, with

$$\|\zeta_{k-1}\| \leq \frac{1}{\sqrt{N}} \|\widetilde{x}^{(c)}(k)\|. \tag{4.61}$$

That is: 1) $\overline{x}(k) \in \mathcal{X}$, and 2) $\|\overline{x}(k) - P_{\mathcal{X}}\left\{\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N} \nabla f_i(y_i(k-1))\right\}\| \leq \zeta_{k-1}$.

Proof: We first prove claim 1 ($\overline{x}(k) \in \mathcal{X}$). Note that $\overline{x}(k) = \frac{1}{N}\sum_{i=1}^{N} P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\}$, and so it belongs to $\mathcal{X}$ as a convex combination of the points that belong to $\mathcal{X}$. We next prove claim 2. Using $\overline{x}^{(c)}(k) = \overline{y}(k-$

$1) - \frac{\alpha}{N}\sum_{i=1}^{N}\nabla f_i(y_i(k-1))$, equations (4.58) and (4.60), and expressing $\overline{x}(k) = \frac{1}{N}\sum_{i=1}^{N} P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\}$:

$$\|\overline{x}(k) - P_{\mathcal{X}}\left\{\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N}\nabla f_i(y_i(k-1))\right\}\| \leq \frac{1}{N}\sum_{i=1}^{N}\|P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\} - P_{\mathcal{X}}\left\{\overline{x}^{(c)}(k)\right\}\|. \quad (4.62)$$

Consider the right hand side in (4.62). Expressing $x_i^{(c)}(k) = \overline{x}^{(c)}(k) + \widetilde{x}_i^{(c)}(k)$, and using the non-expansiveness property of projection: $\|P_{\mathcal{X}}\{u\} - P_{\mathcal{X}}\{v\}\| \leq \|u - v\|, \forall u, v \in \mathbb{R}^d$, obtain:

$$\left\|\overline{x}(k) - P_{\mathcal{X}}\left\{\overline{y}(k-1) - \frac{\alpha}{N}\sum_{i=1}^{N}\nabla f_i(y_i(k-1))\right\}\right\| \leq \frac{1}{N}\sum_{i=1}^{N}\left\|\widetilde{x}_i^{(c)}(k)\right\| \leq \frac{1}{\sqrt{N}}\left\|\widetilde{x}^{(c)}(k)\right\|, \quad (4.63)$$

where the last inequality follows by convexity of $u \mapsto u^2$: $\left(\frac{1}{N}\sum_{i=1}^{N}\|\widetilde{x}_i^{(c)}(k)\|\right)^2 \leq \frac{1}{N}\sum_{i=1}^{N}\|\widetilde{x}_i^{(c)}(k)\|^2 = \frac{1}{N}\|\widetilde{x}^{(c)}\|^2$. □

Disagreement estimate

We next find the bounds on $\|\widetilde{y}(k)\|$ and $\|\widetilde{x}^{(c)}(k)\|$, in order to characterize the oracle inexactness δ_k and the projection inexactness ζ_k.

Lemma 4.17 Consider the projected mD–NC algorithm under Assumptions 4.14 and 4.15, and set the step size $\alpha \leq 1/(2L)$. Then, for all $k = 1, 2, ...$:

$$\left(\mathbb{E}\left[\|\widetilde{x}^{(c)}(k)\|\right]\right)^2 \leq \mathbb{E}\left[\|\widetilde{x}^{(c)}(k)\|^2\right] \leq \frac{N\,(3B + \alpha G)^2}{k^8} \quad (4.64)$$

$$\left(\mathbb{E}\left[\|\widetilde{y}(k)\|\right]\right)^2 \leq \mathbb{E}\left[\|\widetilde{y}(k)\|^2\right] \leq \frac{5\,N\,(3B + \alpha G)^2}{k^8}. \quad (4.65)$$

Proof: The left inequalities in (4.64) and (4.65) follow, e.g., from the Jensen inequality $h\left(\mathbb{E}[Z]\right) \leq \mathbb{E}[h(Z)]$, with $h(z) = z^2$.

We now prove the two right inequalities. We conduct the proof for $d = 1$, while the extension to generic d is straightforward.

The proof has four steps. In Step 1, we upper bound $\|y(k)\|$. In Step 2, we prove (4.64). In Step 3, we upper bound $\mathbb{E}\left[\|\widetilde{x}(k)\|^2\right]$. Finally, in Step 4, we prove (4.65).

Step 1: Bounding $\|y(k)\|$. We first prove a bound for $\|\widetilde{y}(k)\|$. Consider step 3 in Algorithm 3 and fix arbitrary node i. Note that $x_i^{(b)}(k-1)$ belongs to $\mathcal{X}$, because it is a convex combination of the points

$x_j(k-1)$, $j=1,...,N$, that belong to the set $\mathcal{X}$. Next, using $|\beta_{k-1}| \leq 1$:

$$\|y_i(k)\| \leq 2\|x_i(k)\| + \|x_i^{(b)}(k-1)\| \leq 3B,$$

as $x_i(k), x_i^{(b)}(k-1) \in \mathcal{X}$, $\forall k$. Thus, we obtain the desired bound:

$$\|y(k)\| \leq 3\sqrt{N}B, \ \forall k. \tag{4.66}$$

Step 2: Proof of (4.64). Recall the definition of $\mathcal{W}(k)$ in (4.56). and $\widetilde{\mathcal{W}}(k) = \mathcal{W}(k) - J$. From steps 2 and 3 in Algorithm 3, note that $\widetilde{x}^{(c)}(k)$ can be written as:

$$\widetilde{x}^{(c)}(k) = \widetilde{\mathcal{W}}(k)(I-J)\left(y(k-1) - \alpha \nabla F(y(k-1))\right).$$

Recall Lemma 3.13 in Chapter 2. Take the norm, use the sub-multiplicative and sub-additive properties of norms, and square the obtained inequality. Further, use $\|I-J\| = 1$, inequality (4.66), $\|\nabla F(y(k-1))\| \leq \sqrt{N}G$, Lemma 3.13, and the Jensen inequality, to obtain (4.64).

Step 3: Upper bounding $\mathbb{E}\left[\|\widetilde{x}(k)\|^2\right]$. For $\widetilde{x}_i(k) := x_i(k) - \overline{x}(k)$, using $x_i(k) = P_{\mathcal{X}}\left\{x_i^{(c)}(k)\right\}$ and $\overline{x}(k) = \frac{1}{N}\sum_{j=1}^N P_{\mathcal{X}}\{x_j(k)\}$, we have:

$$\begin{aligned}
\|\widetilde{x}_i(k)\| &= \left\| P_{\mathcal{X}}\{x_i^{(c)}(k)\} - \frac{1}{N}\sum_{j=1}^N P_{\mathcal{X}}\{x_j^{(c)}(k)\} \right\| = \left\| \frac{1}{N}\sum_{j=1}^N \left(P_{\mathcal{X}}\{x_i^{(c)}(k)\} - P_{\mathcal{X}}\{x_j^{(c)}(k)\} \right) \right\| \\
&\leq \frac{1}{N}\sum_{j=1}^N \left\| P_{\mathcal{X}}\{x_i^{(c)}(k)\} - P_{\mathcal{X}}\{x_j^{(c)}(k)\} \right\| \leq \frac{1}{N}\sum_{j=1}^N \left\| x_i^{(c)}(k) - x_j^{(c)}(k) \right\| \qquad (4.67) \\
&\leq \frac{1}{N}\sum_{j=1}^N \left(\left\| \widetilde{x}_i^{(c)}(k) \right\| + \left\| \widetilde{x}_j^{(c)}(k) \right\| \right) \leq \left\| \widetilde{x}_i^{(c)}(k) \right\| + \frac{1}{\sqrt{N}} \left\| \widetilde{x}^{(c)}(k) \right\|. \qquad (4.68)
\end{aligned}$$

The left inequality in (4.67) is by convexity of norms, while the right inequality is by the non-expansiveness of the Euclidean projection: $\|P_{\mathcal{X}}\{a\} - P_{\mathcal{X}}\{b\}\| \leq \|a-b\|$, $\forall a, b \in \mathbb{R}^d$. The left inequality in (4.68) is by expressing $\|x_i^{(c)}(k) - x_j^{(c)}(k)\| = \|(x_i^{(c)}(k) - \overline{x}^{(c)}(k)) + (\overline{x}^{(c)}(k) - x_j^{(c)})\| \leq \|x_i^{(c)}(k) - \overline{x}^{(c)}(k)\| + \|\overline{x}^{(c)}(k) - x_j^{(c)})\|$; and the right inequality in (4.68) is by $\left(\frac{1}{N}\sum_{i=1}^N \|\widetilde{x}_j^{(c)}(k)\|\right)^2 \leq \frac{1}{N}\sum_{i=1}^N \|\widetilde{x}_j^{(c)}(k)\|^2 = \frac{1}{N}\|\widetilde{x}^{(c)}(k)\|^2$. Summing the squared right inequalities in (4.68) over $i=1,...,N$, and using

$$\left(\left\| \widetilde{x}_i^{(c)}(k) \right\|^2 + \frac{1}{\sqrt{N}} \left\| \widetilde{x}^{(c)}(k) \right\| \right)^2 \leq 2 \left\| \widetilde{x}_i^{(c)}(k) \right\|^2 + \frac{2}{N} \left\| \widetilde{x}^{(c)}(k) \right\|^2,$$

we obtain:

$$\|\widetilde{x}(k)\|^2 \leq 4\,\|\widetilde{x}^{(c)}(k)\|^2.$$

Thus, from (4.64), we obtain the desired bound:

$$\left(\mathbb{E}\left[\|\widetilde{x}(k)\|\right]\right)^2 \leq \mathbb{E}\left[\|\widetilde{x}^{(c)}(k)\|^2\right] \leq \frac{4\,N(3B+\alpha G)^2}{k^8}. \tag{4.69}$$

Step 4: Proof of (4.65). From step 5 in Algorithm 3, we have:

$$\begin{aligned}\widetilde{y}(k) &= (1+\beta_{k-1})\widetilde{x}(k) - \beta_{k-1}\widetilde{x}^{(b)}(k-1)\\ &= (1+\beta_{k-1})\widetilde{x}(k) - \beta_{k-1}\widetilde{\mathcal{W}}(k)\,(I-J)\,x(k-1).\end{aligned}$$

Thus, using $\|\beta_{k-1}\| \leq 1$:

$$\|\widetilde{y}(k)\| \leq 2\,\|\widetilde{x}(k)\| + \|\widetilde{\mathcal{W}}(k)\|\,\|\widetilde{x}(k-1)\|.$$

Squaring the latter inequality, using $(a+b)^2 \leq 2a^2+2b^2$, (4.69), and $\|(I-J)x(k-1)\| \leq \sqrt{N}B$, we obtain:

$$\|\widetilde{y}(k)\|^2 \leq 4\,\|\widetilde{x}(k)\|^2 + 2\|\widetilde{\mathcal{W}}(k)\|^2\,\|\widetilde{x}(k-1)\|^2.$$

Taking expectations, using (4.69), and using Jensen's inequality, we finally obtain (4.65). □

Convergence rate

We are now ready to state the convergence rate result for the projected mD–NC algorithm.

Theorem 4.18 Consider the projected mD–NC given in Algorithm 3 under Assumptions 3.1, 3.2, and 4.14 with the constant step size $\alpha \leq 1/(2L)$. Let $\|\overline{x}(0)-x^\star\| \leq R$, $R \geq 0$. Then, after

$$\mathcal{K} = \sum_{t=1}^{k} \tau_t \leq \frac{1}{-\log\bar{\mu}}\left(4(k+1)\log(k+1) + (k+1)\log N\right)$$

communication rounds, i.e., after k outer iterations, we have, at any node i:

$$\frac{\mathbb{E}\left[f(x_i(k)) - f^\star\right]}{N} \leq \frac{1}{k^2}\left(\frac{2}{\alpha}R^2 + a_1'\,LB^2 + a_2'\,L(6B+\alpha G)^2 + \alpha\,G^2\right), \quad k=1,2,..., \tag{4.70}$$

where a_1' and a_2' are universal constants independent of system parameters.

Proof: [Proof outline] We apply Lemma 4.13 with $x^\bullet \equiv x^\star$. Further, as $\delta_k \leq L\|\widetilde{y}(k)\|^2$. we have, by Lemma 4.16, and Lemma 4.17: $\delta_k \leq L\|\widetilde{y}(k)\|^2 = \frac{9NLB^2}{k^8}$, $\zeta_k \leq \frac{\|\widetilde{x}^{(b)}(k)\|}{\sqrt{N}} \leq \frac{3B+\alpha G}{k^4}$. Finally, set $L_{k-1} \equiv N/\alpha$, and note that $\|\widehat{g}_k\| = \|\sum_{i=1}^N \nabla f_i(y_i(k))\| \leq NG$, $\forall k$, as $y_i(x) \in \mathcal{X}'$, $\forall k$. We now have all the relevant quantities set, and the proof proceeds by applying Lemma 4.13, and is similar to proofs of Theorem 3.8 and 3.15. □

4.6 Proof of Lemma 4.13

Proof: We perform the proof in three steps.

Step 1. We first prove the following auxiliary equality:

$$\theta_{k-1}\overline{v}(k) = \overline{x}(k) - (1-\theta_{k-1})\overline{x}(k-1). \tag{4.71}$$

Using the definition of $\overline{v}(k)$ in (4.54), $\theta_k = 2/(k+2)$, $\beta_{k-1} = (k-1)/(k+2)$, and (4.52):

$$\overline{v}(k) = \frac{k+2}{2}\left(\overline{x}(k) + \frac{k-1}{k+2}\overline{x}(k) - \frac{k-1}{k+2}\overline{x}(k-1) - \frac{k}{k+2}\overline{x}(k)\right) = \frac{k+1}{2}\overline{x}(k) - \frac{k-1}{2}\overline{x}(k-1).$$

Multiplying the expression on the right hand side of the last equality by $\theta_{k-1} = 2/(k+1)$, the result follows.

Step 2. We prove the following relation:

$$f(\overline{x}(k)) \leq f(z) + L_{k-1}(\overline{x}(k) - \overline{y}(k-1))^\top (z - \overline{x}(k)) + \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 + \delta_{k-1} + \eta_{k-1},\ \forall z \in \mathcal{X}. \tag{4.72}$$

Because $\overline{x}(k) \in \mathcal{X}$ (by construction), we have, using (4.50):

$$f(\overline{x}(k)) \leq \widehat{f}_{k-1} + \widehat{g}_{k-1}^\top (\overline{x}(k) - \overline{y}(k-1)) + \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 + \delta_{k-1}. \tag{4.73}$$

Denote by $p := P_{\mathcal{X}}\left\{\overline{y}(k-1) - \frac{1}{L_{k-1}}\widehat{g}_{k-1}\right\}$. We next upper bound the term

$$\Pi(z) := L_{k-1}\left(\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - \overline{x}(k)\right)^\top (\overline{x}(k) - z)\,,$$

for arbitrary $z \in \mathcal{X}$. Adding and subtracting p in the second and third factors of $\Pi(z)$, obtain:

$$\begin{aligned}\Pi(z) &= L_{k-1}\left(\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\right)^{\top}(p-z) \\ &+ L_{k-1}\left(\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\right)^{\top}(\widehat{x}(k)-p) + L_{k-1}(p-\widehat{x}(k))^{\top}(p-z) - L_{k-1}\|p-\widehat{x}(k)\|^2 \\ &\geq -L_{k-1}\|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\|\,\|\widehat{x}(k)-p\| - L_{k-1}\|p-\widehat{x}(k)\|\,\|p-z\| - L_{k-1}\|p-\widehat{x}(k)\|^2 \quad (4.74)\end{aligned}$$

The inequality follows by: 1) upper bounding the last three summands of $\Pi(z)$ via $u^{\top}v \geq -\|u\|\,\|v\|$, $\forall u, v \in \mathbb{R}^d$; and 2) using the fact that the first summand is nonnegative by the following projection property: $(w - P_{\mathcal{X}}\{w\})^{\top}(p_{\mathcal{X}}\{w\} - z) \geq 0, \forall z \in \mathcal{X}$. We next upper bound $\|\overline{x}(k)-p\|$, $\|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\|$, and $\|p-z\|$. Upper bound $\|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\|$ using the sub-additive property of norms:

$$\|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\| \leq \|\overline{y}(k-1)\| + \frac{\|\widehat{g}_{k-1}\|}{L_{k-1}} + \|p\|.$$

Next, from (4.52), $|\beta_{k-1}| \leq 1$, and because $\overline{x}(k-1), \overline{x}(k) \in \mathcal{X}$: $\|\overline{y}(k-1)\| \leq 3B$. Also, because $\overline{x}(k), p \in \mathcal{X}$, we have $\|\overline{x}(k)\| \leq B$ and $\|p\| \leq B$. Using the latter bounds on $\|\overline{y}(k-1)\|$, $\|\overline{x}(k)\|$, and $\|p\|$:

$$\|\overline{x}(k)-p\| \leq \zeta_{k-1}, \;\; \|\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - p\| \leq 4B + \frac{\|\widehat{g}_{k-1}\|}{L_{k-1}}, \;\; \|z-p\| \leq 2B, \quad (4.75)$$

where the bound on $\|\overline{x}(k)-p\|$ is by the algorithm construction. Applying (4.75) to (4.74), obtain:

$$0 \leq \Pi(z) + \eta_{k-1} = L_{k-1}\left(\overline{y}(k-1) - \frac{\widehat{g}_{k-1}}{L_{k-1}} - \overline{x}(k)\right)^{\top}(\overline{x}(k)-z) + \eta_{k-1}, \quad (4.76)$$

where η_{k-1} is given in (4.55). From property (4.49): $\widehat{f}_{k-1} \leq f(z) + \widehat{g}_{k-1}^{\top}(\overline{y}(k-1)-z)$, and so, using the last equation and adding (A.3) and (A.4), the claim (A.2) follows.

Step 3. We finalize the proof of Lemma 4.13 by proving (4.53). We start by using relation (A.2). Namely: 1) setting $z = \overline{x}(k-1)$ in (A.2) and multiplying inequality (A.2) by $1-\theta_{k-1}$; 2) setting $z = x^{\bullet}$

in (A.2) and multiplying inequality (A.2) by θ_{k-1}; and 3) adding the corresponding two inequalities:

$$\begin{aligned}
& \theta_{k-1}\left\{f(\overline{x}(k))-f(x^{\bullet})\right\}+(1-\theta_{k-1})\left\{f(\overline{x}(k))-f(\overline{x}(k-1))\right\} \\
= \;& \left\{f(\overline{x}(k))-f(x^{\bullet})\right\}-(1-\theta_{k-1})\left\{f(\overline{x}(k-1))-f(x^{\bullet})\right\} \\
\leq \;& \theta_{k-1}L_{k-1}\,\left(\overline{x}(k)-\overline{y}(k-1)\right)^{\top}(x^{\bullet}-\overline{x}(k))+(1-\theta_{k-1})L_{k-1}\,\left(\overline{x}(k)-\overline{y}(k-1)\right)^{\top}(\overline{x}(k-1)-\overline{x}(k)) \\
+ \;& \frac{L_{k-1}}{2}\|\overline{x}(k)-\overline{y}(k-1)\|^{2}+\delta_{k-1}+\eta_{k-1} \\
= \;& L_{k-1}(\overline{x}(k)-\overline{y}(k-1))^{\top}(\theta_{k-1}x^{\bullet}+(1-\theta_{k-1})\overline{x}(k-1)-\overline{x}(k))+\frac{L_{k-1}}{2}\|\overline{x}(k)-\overline{y}(k-1)\|^{2}+\delta_{k-1}+\eta_{k-1} \\
= \;& \frac{L_{k-1}}{2}(2(\overline{x}(k)-\overline{y}(k-1))^{\top}(\theta_{k-1}x^{\bullet}+(1-\theta_{k-1})\overline{x}(k-1)-\overline{x}(k)) \\
+ \;& \|\overline{x}(k)-\overline{y}(k-1)\|^{2})+\delta_{k-1}+\eta_{k-1}. \qquad (4.77)
\end{aligned}$$

Denote by:

$$\mathcal{M}_{k-1}=(2(\overline{x}(k)-\overline{y}(k-1))^{\top}(\theta_{k-1}x^{\bullet}+(1-\theta_{k-1})\overline{x}(k-1)-\overline{x}(k))+\|\overline{x}(k)-\overline{y}(k-1)\|^{2}).$$

Then, inequality (A.5) is written simply as:

$$\left\{f(\overline{x}(k))-f(x^{\bullet})\right\}-(1-\theta_{k-1})\left\{f(\overline{x}(k-1))-f(x^{\bullet})\right\}\leq\frac{L_{k-1}}{2}\mathcal{M}_{k-1}+\delta_{k-1}+\eta_{k-1}. \qquad (4.78)$$

Now, we simplify the expression for $\mathcal{M}_{k-1}$ as follows. Using the identity:

$$\|\overline{x}(k)-\overline{y}(k-1)\|^{2}=2(\overline{x}(k)-\overline{y}(k-1))^{\top}\overline{x}(k)+\|\overline{y}(k-1)\|^{2}-\|\overline{x}(k)\|^{2},$$

we have:

$$\begin{aligned}
\mathcal{M}_{k-1} &= 2(\overline{x}(k)-\overline{y}(k-1))^{\top}(\theta_{k-1}x^{\bullet}+(1-\theta_{k-1})\overline{x}(k-1))-\|\overline{x}(k)\|^{2}+\|\overline{y}(k-1)\|^{2} \\
&= \|\overline{y}(k-1)-((1-\theta_{k-1})\overline{x}(k-1)+\theta_{k-1}x^{\bullet})\|^{2}-\|\overline{x}(k)-((1-\theta_{k-1})\overline{x}(k-1)+\theta_{k-1}x^{\bullet})\|^{2} \\
&= \theta_{k-1}^{2}\|\overline{v}(k-1)-x^{\bullet}\|^{2}-\theta_{k-1}^{2}\|\overline{v}(k)-x^{\bullet}\|^{2}, \qquad (4.79)
\end{aligned}$$

where the last equality follows by the definition of $\overline{v}(k-1)$ in (4.54) and by the identity (A.1). Now, combining (A.6) and (A.7):

$$\begin{aligned}
(f(\overline{x}(k))-f(x^{\bullet})) &- (1-\theta_{k-1})(f(\overline{x}(k-1))-f(x^{\bullet})) \\
&\leq \frac{L_{k-1}\theta_{k-1}^{2}}{2}\left(\|\overline{v}(k-1)-x^{\bullet}\|^{2}-\|\overline{v}(k)-x^{\bullet}\|^{2}\right)+\delta_{k-1}+\eta_{k-1}.
\end{aligned}$$

Finally, multiplying the last equation by $\frac{4}{\theta_{k-1}^2}$, and using $\theta_{k-1} = 2/(k+1)$, we get the result. □

4.7 Simulation example

We provide a simulation example for the D–NG method and an acoustic source localization problem. The cost function specified further ahead does not have bounded gradients, but it satisfies Assumptions 4.2 and 4.3, i.e., it fall in the setup of Section 4.3.

We explain the source localization problem. A sensor network instruments the environment where an acoustic source is positioned at an unknown location $\theta \in \mathbb{R}^2$, e.g. [64]. The source emits a signal isotropically. Each node (sensor) i measures the received signal energy:

$$y_i = \frac{A}{\|\theta - r_i\|^\kappa} + \zeta_i. \tag{4.80}$$

Here $r_i \in \mathbb{R}^2$ is node i's location, known to node i, $A > 0$ and $\kappa > 0$ are constants known to all nodes, and ζ_i is zero-mean additive noise. The goal is for each node to estimate the source's position θ. A straightforward approach is to find the nonlinear least squares estimate $\overline{\theta} = x^\star$ by minimizing the following cost function (of the variable x):

$$\text{minimize} \quad \sum_{i=1}^N \left(y_i - \tfrac{A}{\|x-r_i\|^\kappa}\right)^2 . \tag{4.81}$$

Problem (4.81) is nonconvex and is difficult; still, it is possible to efficiently obtain a good estimator $\widehat{\theta}$ based on the data y_i, $i = 1, ..., N$, by solving the following *convex* problem:

$$\text{minimize} \ \sum_{i=1}^N \operatorname{dist}^2(x, C_i), \tag{4.82}$$

where C_i is the disk $C_i = \left\{x \in \mathbb{R}^2 : \|x - r_i\| \le \left(\frac{A}{y_i}\right)^{1/\kappa}\right\}$, and $\operatorname{dist}(x, C) = \inf_{y \in C} \|x - y\|$ is the distance from x to the set C. In words, (4.82) finds a point $\widehat{\theta}$ that has the minimal total squared distance from disks C_i, $i = 1, ..., N$.

The simulation setup is as follows. Each node i acquires a single data sample y_i according to model (4.80). The coefficients $A = 1$ and $\kappa = 2$; the true source's position is $(0.2, 0.2)^\top$; and the measurement noise ζ_i is zero mean, Gaussian, i.i.d. across sensors, with the standard deviation 0.5. In case that the measurement y_i is negative (due to adding a large negative noise ζ_i, we set $y_i = 0$.

The network has $N = 70$ nodes (sensors) and 299 links and is modeled as a geometric graph. Sensors are deployed uniformly randomly on a unit square, and the sensor pairs whose distance is less than a radius

are connected by an (undirected) edge.

Figure 4.1 plots the relative error averaged across nodes $\left(= \frac{1}{N f^\star} \sum_{i=1}^N (f(x_i) - f^\star)\right)$, $f^\star \neq 0$, versus the iteration number k in a $\log_{10} - \log_{10}$ scale. We compare D–NG in (4.7)–(4.8) with the algorithm in [2]. With (4.7)–(4.8), we set the step-size $\alpha_k = 1/(k+1)$; with [2], we set $\alpha_k = 1/[(k+1)^\tau]$, with $\tau \in \{1/10, 1/3, 1/2, 1\}$. We can see that our D–NG method converges much faster in k than the algorithm in [2] for any of the considered step-size choices (choices of τ). For example, for the target average relative error of 0.001, D–NG takes about 500 iterations, while [2] requires about $14,000$ iterations. At the same time, both algorithms have the same communication cost per k and a similar computational cost per k. Also, from Figure 4.1, the rate of convergence (the slope of the log-log plot) is approximately $1/k^2$ with our method (4.7)–(4.8), while the best rate with [2] (among all considered choices of τ) is for $\tau = 1/2$ and is slightly worse than $1/k$.

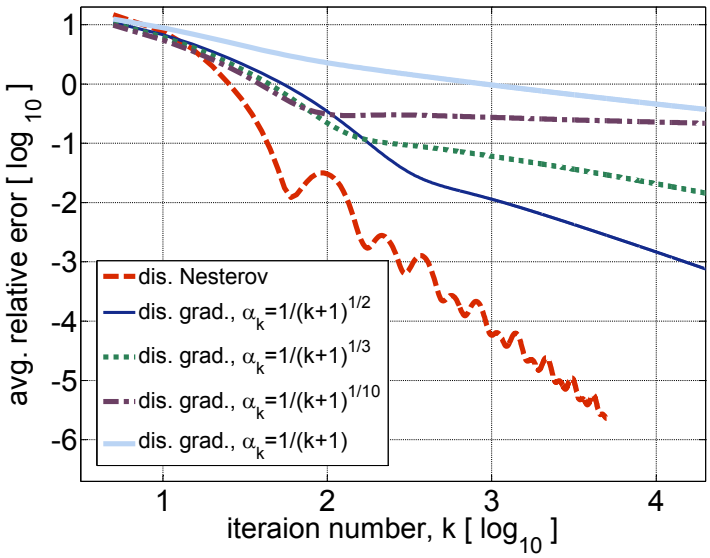

Figure 4.1: Acoustic source localization in sensor networks: Relative error averaged across nodes: $\frac{1}{N f^\star} \sum_{i=1}^N (f(x_i) - f^\star)$, $f^\star \neq 0$, versus iteration number k in a $\log_{10} - \log_{10}$ scale for: 1) algorithm (4.7)–(4.8) with step size $\alpha_k = 1/(k+1)$; and 2) algorithm in [2] with $\alpha_k = 1/(k+1)^\tau$, $\tau \in \{1/10, 1/3, 1/2, 1\}$.

4.8 Conclusion

In this Chapter, we analyzed our distributed Nesterov-like gradient methods under alternative function classes, in the following three scenarios. The first two scenarios holds for static networks, while the third scenario is valid for random networks as well. 1) We analyze D–NG method for unconstrained problems,

when the bounded gradients Assumption is replaced with a certain growth condition and show that rates $O(\log k/k)$ and $O(\log \mathcal{K}/\mathcal{K})$ are still achieved by the D–NG. 2) We analyze projected D–NG method for constrained optimization and differentiable, coercive costs with Lipschitz continuous gradients. for compact constraints, we show that the constant step size projected D–NG method converges within ϵ neighborhood of the optimal cost function in $O(1/\epsilon)$ communications and gradient evaluations per-node. Finally, 3) we analyze the projected mD–NC method for differentiable costs with Lipschitz continuous gradients, when the constraint ste is compact, and show $O(1/k^2)$ and $O(1/\mathcal{K}^{2-xi})$ convergence rates.

Chapter 5

Weight Optimization for Consensus over Random Networks

5.1 Introduction

In Chapters 2 and 3, we have seen that the consensus algorithm and the products of stochastic matrices $W(k)$ play an important role in the performance of distributed optimization algorithms like D–NG, D–NC, and their modifications mD–NG and mD–NC. In particular, the convergence constant of these methods depends on the inverse spectral gap $\frac{1}{1-\overline{\mu}}$, where $\overline{\mu} := \left(\|\mathbb{E}\left[W(k)^2\right] - J\|\right)^{1/2}$. The quantity $\overline{\mu}$ depends significantly on the weights that nodes assign to their neighbors. In this Chapter, we address the weight optimization (to minimize $\overline{\mu}$), when the underlying network is random, with spatially correlated and temporally independent links. For simplicity of presentation, we introduce our results from the perspective of the consensus algorithm – distributed computation of an average. However, studying the random consensus dynamics is equivalent to studying the products $W(k)W(k-1)...W(0)$ of the random stochastic matrices. Hence, the results from this Chapter are of directed use in distributed optimization algorithms like mD–NG and mD–NC. We demonstrate this by a simulation example in Section 5.5.

Specifically, we consider the following problem: how to assign the weights C_{ij} with which the nodes mix their states across the network, so that the convergence of the consensus algorithm is the fastest possible. This problem has not been solved (with full generality) for consensus in random topologies. We study this problem for networks with symmetric and asymmetric random links separately, since the properties of the corresponding algorithm are different. For symmetric links (and connected network topology on average), the consensus algorithm converges to the average of the initial nodes' states almost surely. For asymmetric

random links, all the nodes asymptotically reach agreement, but they only agree to a random variable in the neighborhood of the true initial average.

We refer to our weight solution as probability-based weights (PBW). PBW are simple and suitable for distributed implementation: we assume at each iteration that the weight of link (i, j) is C_{ij} (to be optimized), when the link is alive, or 0, otherwise. Self-weights are adapted such that the row-sums of the weight matrix at each iteration are one. This is suitable for distributed implementation. Each node updates readily after receiving messages from its current neighbors. No information about the number of nodes in the network or the neighbor's current degrees is needed. Hence, no additional online communication is required for computing weights, in contrast, for instance, to the case of the Metropolis weights (MW) [65].

Our weight design method assumes that the link occurrence probabilities and their spatial correlations are known. With randomized protocols, the link occurrence probabilities and their correlations are induced by the protocol itself, and thus are known. For networks with random link failures, the link occurrence probabilities relate to the signal to noise ratio at the receiver and can be computed. In [13], the occurrence probabilities are designed in the presence of link communication costs and an overall network communication cost budget. When the WSN infrastructure is known, it is possible to estimate the link occurrence *probabilities* by measuring the reception rate of a link computed as the ratio between the number of received and the total number of sent packets. Another possibility is to estimate the link occurrence probabilities based on the received signal strength. Link occurrence *correlations* can also be estimated on actual WSNs, [83]. If there is no training period to characterize quantitatively the links on an actual WSN, we can still model the probabilities and the correlations as a function of the transmitted power and the inter-sensor distances. Moreover, several empirical studies ([83, 84] and references therein) on the quantitative properties of wireless communication in sensor networks have been done that provide models for packet delivery performance in WSNs.

Contribution. Building our results on the previous extensive studies of convergence conditions and rates for consensus algorithm, e.g., [1, 80, 13], we address optimization of the weights in consensus algorithms with correlated random topologies. Our method is applicable to: 1) networks with correlated random link failures (see, e.g., [13] and 2) networks with randomized algorithms (see, e.g, [10, 1]). We first address the weight design problem for symmetric random links and then extend the results to asymmetric random links.

With symmetric random links, we use as the optimization criterion the mean squared consensus convergence rate $\overline{\mu}^2$, which we denote here by $\phi(C)$, thus indicating the dependence on the weights C. We explicitly express the rate $\phi(C)$ as a function of the link occurrence probabilities, their correlations, and the weights. We prove that $\phi(C)$ is a convex, nonsmooth function of the weights. This enables global optimiza-

tion of the weights for arbitrary link occurrence probabilities and and arbitrary link correlation structures. We solve numerically the resulting optimization problem by a subgradient algorithm, showing also that the optimization computational cost grows tolerably with the network size. We provide insights into weight design with a simple example of complete random network that admits a closed form solution for the optimal weights and convergence rate and show how the optimal weights depend on the number of nodes, the link occurrence probabilities, and their correlations.

We extend our results to the case of asymmetric random links, adopting as an optimization criterion the mean squared deviation (from the current average state) rate $\psi(C)$, and show that $\psi(C)$ is a convex function of the weights.

We provide simulation experiments to demonstrate the effectiveness of our approach. We provide two different models of random networks with correlated link failures; in addition, we study the broadcast gossip algorithm [1], as an example of randomized protocol with asymmetric links. In all cases, simulations confirm that our method shows significant gain compared to the methods available in the literature. Finally, we demonstrate by a simulation example with the mD–NG algorithm in Chapter 3 and Huber loss cost functions that the PBW can significantly improve convergence constant with distributed optimization algorithms.

Related work. Reference [13] studies the tradeoff between the convergence rate and the amount of communication that takes place in the network. This reference is mainly concerned with the design of the network topology, i.e., the design of the probabilities of reliable communication $\{P_{ij}\}$ and the weight α (assuming all nonzero weights are equal), assuming a communication cost C_{ij} per link and an overall network communication budget. Reference [1] proposes the broadcast gossip algorithm, where at each time step, a single node, selected at random, broadcasts unidirectionally its state to all the neighbors within its wireless range. We detail the broadcast gossip in subsection 5.5.2. This reference optimizes the weight for the broadcast gossip algorithm assuming equal weights for all links.

The problem of optimizing the weights for consensus under a random topology, when the weights for different links may be different, has not received much attention in the literature. Authors have proposed weight choices for random or time-varying networks [85, 65], but no claims to optimality are made. Reference [65] proposes the Metropolis weights (MW), based on the Metropolis-Hastings algorithm for simulating a Markov chain with uniform equilibrium distribution [86]. The weights choice in [85] is based on the fastest mixing Markov chain problem studied in [87] and uses the information about the underlying supergraph. We refer to this weight choice as the supergraph based weights (SGBW).

Summary of the chapter. Section 5.2 describes our model of random networks and the consensus algorithm. Sections 5.3 and 5.4 study the weight optimization for symmetric random graphs and asymmetric

random graphs, respectively. Section 5.5 demonstrates the effectiveness of our approach with simulations. Finally, Section 5.6 concludes the chapter. We derive the proofs of some results in Appendices C.

Notation. Symbol $\mathbb{R}^N$ is the N-dimensional Euclidean space. Inequality $x \leq y$ is understood element wise, i.e., it is equivalent to $x_i \leq y_i$, for all i. A sequence of random matrices is denoted by $\{W(k)\}_{k=0}^{\infty}$ and the random matrix indexed by k is denoted $W(k)$. If the distribution of $W(k)$ is the same for any k, we shorten the notation $W(k)$ to W when the time instant k is not of interest. Symbol $\mathbb{R}^{N\times M}$ denotes the set of $N \times M$ real valued matrices and $\mathbb{S}^N$ denotes the set of symmetric real valued $N \times N$ matrices. The i-th column of a matrix W is denoted by W_i. Matrix entries are denoted by W_{ij}. Quantities $W \otimes V$, $W \odot V$, and $W \oplus V$ denote the Kronecker product, the Hadamard product, and the direct sum of the matrices W and V, respectively. Inequality $W \succeq V$ ($W \preceq V$) means that the matrix $W - V$ is positive (negative) semidefinite. Inequality $W \geq V$ ($W \leq V$) is understood entry wise, i.e., it is equivalent to $W_{ij} \geq V_{ij}$, for all i, j. Quantities $\|W\|$, $\lambda_i(W)$, and $r(W)$ denote the matrix 2-norm, the i-th smallest eigenvalue, and the spectral radius of W, respectively. The identity matrix is I. Given a matrix W, Vec(W) is the column vector that stacks the columns of W. For given scalars $x_1, ..., x_N$, $\mathrm{diag}\,(x_1, ..., x_N)$ denotes the diagonal $N \times N$ matrix with the i-th diagonal entry equal to x_i. Similarly, $\mathrm{diag}(x)$ is the diagonal matrix whose diagonal entries are the elements of x. The matrix $\mathrm{diag}\,(W)$ is a diagonal matrix with the diagonal equal to the diagonal of W. The N-dimensional column vector of ones is denoted by 1. The ideal consensus matrix $J = \frac{1}{N}11^\top$. The i-th canonical unit vector, i.e., the i-th column of I, is denoted by e_i. Symbol $|S|$ denotes the cardinality of a set S.

The result in this Chapter have been published in [46].

5.2 Problem model

This section introduces the random network model that we apply to networks with link failures and to networks with randomized algorithms. It also introduces the consensus algorithm and the corresponding weight rule assumed in this chapter.

5.2.1 Random network model: symmetric and asymmetric random links

We consider random networks−networks with random links or with a random protocol. Random links arise because of packet loss or drop, or when a sensor is activated from sleep mode at a random time. Randomized protocols like standard pairwise gossip [10] or broadcast gossip [1] activate links randomly. This section describes the network model that applies to both problems. We assume that the links are up or down (link

failures) or selected to use (randomized gossip) according to spatially correlated Bernoulli random variables.

To be specific, the network is modeled by a graph $G = (V, E)$, where the set of nodes V has cardinality $|V| = N$ and the set of directed edges E, with $|E| = 2M$, collects all possible ordered node pairs that can communicate, i.e., all realizable links. For example, with geometric graphs, realizable links connect nodes within their communication radius. The graph G is called supergraph, e.g., [13]. The directed edge $(i, j) \in E$ if node j can transmit to node i.

The supergraph G is assumed to be connected and without loops. For the fully connected supergraph, the number of directed edges (arcs) $2M$ is equal to $N(N-1)$. We are interested in sparse supergraphs, i.e., the case when $M \ll \frac{1}{2}N(N-1)$.

Associated with the graph G is its $N \times N$ adjacency matrix A:

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

The in-neighborhood set Ω_i (nodes that can transmit to node i) and the in-degree d_i of a node i are

$$\begin{aligned} \Omega_i &= \{j : (i,j) \in E\} \\ d_i &= |\Omega_i|. \end{aligned}$$

We model the connectivity of a random WSN at time step k by a (possibly) directed random graph $\mathcal{G}(k) = (V, \mathcal{E}(k))$. The random edge set is

$$\mathcal{E}(k) = \{(i,j) \in E : \ (i,j) \text{ is online at time step k}\},$$

with $\mathcal{E}(k) \subseteq E$. The random adjacency matrix associated to $\mathcal{G}(k)$ is denoted by $\mathcal{A}(k)$ and the random in-neighborhood for sensor i by $\Omega_i(k)$.

We assume that link failures are *temporally independent* and *spatially correlated*. That is, we assume that the random matrices $\mathcal{A}(k), k = 0, 1, 2, ...$ are independent identically distributed. The state of the link (i, j) at a time step k is a Bernoulli random variable, with mean P_{ij}, i.e., P_{ij} is the occurrence probability of link (i, j). At time step k, different edges (i, j) and (p, q) may be correlated, i.e., the entries $\mathcal{A}_{ij}(k)$ and $\mathcal{A}_{pq}(k)$ may be correlated. For the link r, by which node j transmits to node i, and for the link s, by which

node q transmits to node p, the corresponding cross-variance is

$$[R_q]_{rs} = \mathbb{E}\left[\mathcal{A}_{ij}\mathcal{A}_{pq}\right] - P_{ij}P_{pq}.$$

5.2.2 Consensus algorithm

We recall the consensus algorithm from Chapter 2. Besides the consensus problem (distributed computation of an average), the random consensus dynamics plays an important role in the performance of distributed optimization algorithms like D–NG and D–NC in Chapter 2 and their modifications mD–NG and mD–NC in Chapter 3. In this Chapter, we present the results mainly in the context of consensus algorithm; Section 5.5 considers a simulation example on distributed optimization via mD–NG.

Let $x_i(0)$ represent a scalar measurement or initial data available at sensor i, $i = 1, ..., N$. Denote by x_{avg} the average:

$$x_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N} x_i(0).$$

The consensus algorithm computes x_{avg} iteratively at each sensor i by the distributed weighted average:

$$x_i(k+1) = W_{ii}(k)x_i(k) + \sum_{j\in\Omega_i(k)} W_{ij}(k)x_j(k), \;\; k = 0, 1, ... \tag{5.1}$$

We assume that the random weights $W_{ij}(k)$ at iteration k are given by:

$$W_{ij}(k) = \begin{cases} C_{ij} & \text{if } j \in \Omega_i(k) \\ 1 - \sum_{m\in\Omega_i(k)} W_{im}(k) & \text{if } i = m \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

In (5.2), the quantities C_{ij} are non random and will be the variables to be optimized in this Chapter. We also take $C_{ii} = 0$, for all i. By (5.2), when the link is active, the weight is C_{ij}, and when not active it is zero. Note that C_{ij} are non zero only for edges (i, j) in the supergraph G. If an edge (i, j) is not in the supergraph the corresponding $C_{ij} = 0$ and $W_{ij}(k) \equiv 0$.

We write the consensus algorithm in compact form. Let $x(k) = (x_1(k)\ x_2(k)\ ...\ x_N(k))^\top$, $W = [C_{ij}]$, $W(k) = [W_{ij}(k)]$. The random weight matrix $W(k)$ can be written in compact form as

$$W(k) = C \odot \mathcal{A}(k) - \text{diag}\left(W\mathcal{A}(k)\right) + I \tag{5.3}$$

and the consensus algorithm is simply stated with $x(k=0)=x(0)$ as

$$x(k+1)=W(k)x(k),\ k\geq 0 \tag{5.4}$$

To implement the update rule, nodes need to know their random in-neighborhood $\Omega_i(k)$ at every iteration. In practice, nodes determine $\Omega_i(k)$ based on who they receive messages from at iteration k.

It is well known [1, 88] that, when the random matrix $W(k)$ is symmetric, the consensus algorithm is average preserving, i.e., the sum of the states $x_i(k)$, and so the average state over time, does not change, even in the presence of random links. In that case the consensus algorithm converges almost surely to the true average x_{avg}. When the matrix $W(k)$ is not symmetric, the average state is not preserved in time, and the state of each node converges to the same random variable with bounded mean squared error from x_{avg} [1].

When the physical communication channels are asymmetric, and the error on the asymptotic consensus limit c is tolerable, consensus with an asymmetric weight matrix $W(k)$ can be used. An example of such a protocol is the broadcast gossip algorithm proposed in [1]. Section 5.4 studies this type of algorithms.

Set of possible weight choices: symmetric network. With symmetric random links, we will always assume $C_{ij}=C_{ji}$. By doing this we easily achieve the desirable property that $W(k)$ is symmetric. The set of all possible weight choices for symmetric random links $\mathcal{C}$ becomes:

$$\mathcal{C}=\left\{C\in\mathbb{R}^{N\times N}:\ C_{ij}=C_{ji},\ C_{ij}=0,\ \text{if}\ (i,j)\notin E,\ C_{ii}=0,\ \forall i\right\} \tag{5.5}$$

Set of possible weight choices: asymmetric network. With asymmetric random links, there is no good reason to require that $C_{ij}=C_{ji}$, and thus we drop the restriction $C_{ij}=C_{ji}$. The set of possible weight choices in this case becomes:

$$\mathcal{C}^{\text{asym}}=\left\{C\in\mathbb{R}^{N\times N}:\ C_{ij}=0,\ \text{if}\ (i,j)\notin E,\ C_{ii}=0,\ \forall i\right\} \tag{5.6}$$

Depending whether the random network is symmetric or asymmetric, there will be two error quantities that will play a role. We introduce them here briefly, for reference.

Mean square error (MSE): symmetric network. Define the consensus error vector $e(k)$ and the error covariance matrix $\Sigma(k)$:

$$e(k) = x(k)-x_{\text{avg}}1 \tag{5.7}$$

$$\Sigma(k) = \mathbb{E}\left[e(k)e(k)^{\top}\right]. \tag{5.8}$$

The mean squared consensus error MSE is given by:

$$\mathrm{MSE}(k) = \sum_{i=1}^{N} \mathbb{E}\left[\left(x_i(k) - x_{\text{avg}}\right)^2\right] = \mathbb{E}\left[e(k)^\top e(k)\right] = \operatorname{tr}\Sigma(k). \tag{5.9}$$

Mean square deviation (MSdev): asymmetric network. As explained, when the random links are asymmetric (i.e., when $W(k)$ is not symmetric), and if the underlying supergraph is strongly connected, then the states of all nodes converge to a common value c that is in general a random variable that depends on the sequence of network realizations and on the initial state $x(0)$ (see [88, 1]). In order to have $c = x_{\text{avg}}$, almost surely, an additional condition must be satisfied:

$$1^\top W(k) = 1^\top, \text{ a.s.} \tag{5.10}$$

See [88, 1] for the details. We remark that (5.10) is a crucial assumption in the derivation of the MSE decay (5.25). Theoretically, equation (5.23) is still valid if the condition $W(k) = W(k)^\top$ is relaxed to $1^\top W(k) = 1^\top$. While this condition is trivially satisfied for symmetric links and symmetric weights $C_{ij} = C_{ji}$, it is very difficult to realize (5.10) in practice when the random links are asymmetric. So, in our work, we do not assume (5.10) with asymmetric links.

For asymmetric networks, we follow reference [1] and introduce the mean square state deviation MSdev as a performance measure. Denote the current average of the node states by $x_{\text{avg}}(k) = \frac{1}{N}1^\top x(k)$. Quantity MSdev describes how far apart different states $x_i(k)$ are; it is given by

$$\mathrm{MSdev}(k) = \sum_{i=1}^{N} \mathbb{E}\left[(x_i(k) - x_{\text{avg}}(k))^2\right] = \mathbb{E}\left[\zeta(k)^\top \zeta(k)\right],$$

where

$$\zeta(k) = x(k) - x_{\text{avg}}(k)1 = (I - J)x(k). \tag{5.11}$$

5.2.3 Symmetric links: Statistics of $W(k)$

In this subsection, we derive closed form expressions for the first and the second order statistics on the random matrix $W(k)$. Let $q(k)$ be the random vector that collects the non redundant entries of $\mathcal{A}(k)$:

$$q_l(k) = \mathcal{A}_{ij}(k),\ i < j,\ (i,j) \in E, \tag{5.12}$$

where the entries of $\mathcal{A}(k)$ are ordered in lexicographic order with respect to i and j, from left to right, top to bottom. For symmetric links, $\mathcal{A}_{ij}(k) = \mathcal{A}_{ji}(k)$, so the dimension of $q(k)$ is half of the number of directed links, i.e., M. We let the mean and the covariance of $q(k)$ and $\mathrm{Vec}\,(\mathcal{A}(k))$ be:

$$\pi = \mathbb{E}\,[q(k)] \tag{5.13}$$

$$\pi_l = \mathbb{E}[q_l(k)] \tag{5.14}$$

$$R_q = \mathrm{Cov}(q(k)) = \mathbb{E}[\,(q(k)-\pi)\;(q(k)-\pi)^\top\,] \tag{5.15}$$

$$R_A = \mathrm{Cov}(\,\mathrm{Vec}(\mathcal{A}(k))\,) \tag{5.16}$$

The relation between R_q and R_A can be written as:

$$R_A = F R_q F^\top. \tag{5.17}$$

where $F \in \mathbb{R}^{N^2 \times M}$ is the zero one selection matrix that linearly maps $q(k)$ to $\mathrm{Vec}\,(\mathcal{A}(k))$, i.e., $\mathrm{Vec}\,(\mathcal{A}(k)) = Fq(k)$. We introduce further notation. Let P be the matrix of the link occurrence probabilities

$$P = [P_{ij}]\,.$$

Define the matrix $B \in \mathbb{R}^{N^2 \times N^2}$ with $N \times N$ zero diagonal blocks and $N \times N$ off diagonal blocks B_{ij} equal to:

$$B_{ij} = 1{e_i}^\top + e_j 1^\top.$$

and write C in terms of its columns $C = [C_1\ C_2\ ...\ C_N]$. We let

$$\widehat{C} = C_1 \oplus C_2 \oplus ... \oplus C_N.$$

For symmetric random networks, the mean of the random weight matrix $W(k)$ and of $W^2(k)$ play an important role for the convergence rate of the consensus algorithm. Using the above notation, we can get compact representations for these quantities, as provided in Lemma 1 proved in Appendix C.

Lemma 5.1 Consider the consensus algorithm (5.4). Then the mean and the second moment R_{corr} of W

defined below are:

$$\begin{aligned} \overline{W} &= \mathbb{E}[W] = C \odot P + I - \operatorname{diag}(CP) && (5.18) \\ R_{\text{corr}} &= \mathbb{E}\left[W^2\right] - \overline{W}^2 && (5.19) \\ &= \widehat{C}^\top \left\{ R_A \odot (I \otimes 11^\top + 11^\top \otimes I - B) \right\} \widehat{C} && (5.20) \end{aligned}$$

In the special case of spatially uncorrelated links, the second moment R_{corr} of W are

$$\frac{1}{2} R_{\text{corr}} = \operatorname{diag}\left\{ \left(\left(11^\top - P \right) \odot P \right) (C \odot C) \right\} - \left(11^\top - P \right) \odot P \odot C \odot C. \qquad (5.21)$$

For asymmetric random links, the expression for the mean of the random weight matrix $W(k)$ remains the same (as in Lemma 1). For asymmetric random links, instead of $\mathbb{E}\left[W^2\right] - J$, the quantity of interest becomes $\mathbb{E}\left[W^\top (I - J) W\right]$ (The quantity of interest is different since the optimization criterion will be different.) For symmetric links, the matrix $\mathbb{E}\left[W^2\right] - J$ is a quadratic matrix function of the weights C_{ij}; it depends also quadratically on the P_{ij}'s and is an affine function of $[R_q]_{ij}$'s. The same will still hold for $\mathbb{E}\left[W^\top (I - J) W\right]$ in the case of asymmetric random links. The difference, however, is that $\mathbb{E}\left[W^\top (I - J) W\right]$ does not admit the compact representation as given in (5.19), and we do not pursue here cumbersome entry wise representations. In the Appendix C, we do present the expressions for the matrix $\mathbb{E}\left[W^\top (I - J) W(k)\right]$ for the broadcast gossip algorithm [1] (that we study in subsection 5.5.2).

5.3 Weight optimization: symmetric random links

5.3.1 Optimization criterion: Mean square convergence rate

We are interested in finding the rate at which $\text{MSE}(k)$ decays to zero and to optimize this rate with respect to the weights C. First we derive the recursion for the error $e(k)$. We have from (5.4):

$$\begin{aligned} 1^\top x(k+1) &= 1^\top W(k) x(k) = 1^\top x(k) = 1^\top x(0) = N\, x_{\text{avg}} \\ 1^\top e(k) &= 1^\top x(k) - 1^\top 1\, x_{\text{avg}} = N\, x_{\text{avg}} - N\, x_{\text{avg}} = 0 \end{aligned}$$

We derive the error vector dynamics:

$$e(k+1) = x(k+1) - x_{\text{avg}}\, 1 = W(k) x(k) - W(k)\, x_{\text{avg}}\, 1 = W(k) e(k) = (W(k) - J)\; e(k) \qquad (5.22)$$

where the last equality holds because $Je(k) = \frac{1}{N}11^\top e(k) = 0$.

Recall the definition of the mean squared consensus error (5.9) and the error covariance matrix in (5.8) and recall that $\mathrm{MSE}(k) = \mathrm{tr}\,\Sigma(k) = \mathbb{E}\left[e(k)e(k)^\top\right]$. Introduce the quantity

$$\phi(C) = \lambda_N\left(\mathbb{E}[W^2] - J\right). \tag{5.23}$$

The quantity $\phi = \phi(C)$ is precisely the square of the quantity $\overline{\mu}$ in Chapter 3. The next Lemma shows that the mean squared error decays at the rate $\phi(C)$.

Lemma 5.2 (m.s.s convergence rate) Consider the consensus algorithm given by (5.4). Then:

$$\mathrm{tr}\left(\Sigma(k+1)\right) = \mathrm{tr}\left(\left(\mathbb{E}[W^2] - J\right)\Sigma(k)\right) \tag{5.24}$$

$$\mathrm{tr}\left(\Sigma(k+1)\right) \leq \left(\phi(C)\right)\ \mathrm{tr}\left(\Sigma(k)\right),\ k \geq 0. \tag{5.25}$$

From the definition of the covariance $\Sigma(k+1)$, using the dynamics of the error $e(k+1)$, interchanging expectation with the tr operator, using properties of the trace, interchanging the expectation with the tr once again, using the independence of $e(k)$ and $W(k)$, and, finally, noting that $W(k)J = J$, we get (5.24). The independence between $e(k)$ and $W(k)$ follows because $W(k)$ is an i.i.d. sequence, and $e(k)$ depends on $W(0)$,..., $W(k-1)$. Then $e(k)$ and $W(k)$ are independent by the disjoint block theorem [89]. Having (5.24), (5.25) can be easily shown, for example, by exercise 18, page 423, [90].

We remark that, in the case of asymmetric random links, MSE does not asymptotically go to zero. For the case of asymmetric links, we use different performance metric. This will be detailed in section 5.4.

5.3.2 Symmetric links: Weight optimization problem formulation

We now formulate the weight optimization problem as finding the weights C_{ij} that optimize the mean squared rate of convergence:

$$\begin{aligned} &\text{minimize} \quad \phi(C) \\ &\text{subject to} \quad W \in \mathcal{C}. \end{aligned} \tag{5.26}$$

The set $\mathcal{C}$ is defined in (5.6) and the rate $\phi(C)$ is given by (5.23). The optimization problem (5.26) is unconstrained, since effectively the optimization variables are $C_{ij} \in \mathbb{R}$, $(i,j) \in E$, other entries of C being zero.

A point $C^\bullet \in \mathcal{C}$ such that $\phi(C^\bullet) < 1$ will always exist if the supergraph G is connected. Reference [80] studies the case when the random matrices $W(k)$ are stochastic and shows that $\phi(C^\bullet) < 1$ if the supergraph

is connected and all the realizations of the random matrix $W(k)$ are stochastic symmetric matrices. Thus, to locate a point $C^\bullet \in \mathcal{C}$ such that $\phi(C^\bullet) < 1$, we just take $C^\bullet$ that assures all the realizations of W be symmetric stochastic matrices. It is trivial to show that for any point in the set

$$S_{\text{stoch}} = \{C \in \mathcal{C} : \ C_{ij} > 0, \ \text{if } (i,j) \in \mathbb{E}, \ W1 < 1\} \subseteq \mathcal{C} \tag{5.27}$$

all the realizations of $W(k)$ are stochastic, symmetric. Thus, for any point $C^\bullet \in S_{\text{stoch}}$, we have that $\phi(C^\bullet) < 1$ if G is connected.

We remark that the optimum $C^\star$ does not have to lie in the set S_{stoch}. In general, $C^\star$ lies in the set

$$S_{\text{conv}} = \{C \in \mathcal{C} : \phi(C) < 1\} \subseteq \mathcal{C} \tag{5.28}$$

The set S_{stoch} is a proper subset of S_{conv} (If $W \in S_{\text{stoch}}$ then $\phi(C) < 1$, but the converse statement is not true in general.) We also remark that the consensus algorithm (5.4) converges *almost surely* if $\phi(C) < 1$ (not only in mean squared sense). This can be shown, for instance, by the technique developed in [80].

We now relate (5.26) to reference [91]. This reference studies the weight optimization for the case of a $static$ topology. In this case the topology is deterministic, described by the supergraph G. The link occurrence probability matrix P reduces to the supergraph adjacency (zero-one) matrix A, since the links occur always if they are realizable. Also, the link covariance matrix R_q becomes zero. The weight matrix W is deterministic and equal to

$$W \ = \ \overline{W} = \text{diag}\,(CA) - C \odot A + I$$

Recall that $r(X)$ denotes the spectral radius of X. Then, the quantities $\left(r\left(W - J\right)\right)^2$ and $\phi\left(W\right)$ coincide. Thus, for the case of static topology, the optimization problem (5.26) that we address reduces to the optimization problem proposed in [91].

5.3.3 Convexity of the weight optimization problem

We show that $\phi : \mathcal{C} \to \mathbb{R}_+$ is convex, where $\mathcal{C}$ is defined in (5.6) and $\phi(C)$ by (5.23).

Lemma 5.1 gives the closed form expression of $\mathbb{E}\left[W^2\right]$. We see that $\phi(C)$ is the concatenation of a quadratic matrix function and $\lambda_N(\cdot)$. This concatenation is not convex in general. However, the next Lemma shows that $\phi(C)$ is convex for our problem.

Lemma 5.3 (Convexity of $\phi(C)$) The function $\phi : \mathcal{C} \to \mathbb{R}_+$ is convex.

Choose arbitrary $X,\, Y \in \mathcal{C}$. We restrict our attention to matrices C of the form

$$C = X + t\, Y,\, t \in \mathbb{R}. \tag{5.29}$$

Recall the expression for W given by (5.2) and (5.4). For the matrix C given by (5.29), we have for $W = W(t)$

$$\begin{aligned} W(t) &= I - \operatorname{diag}\left[(X + tY)\, \mathcal{A}\right] + (X + tY) \odot \mathcal{A} \qquad (5.30)\\ &= \mathcal{X} + t\mathcal{Y},\, \mathcal{X} = X \odot \mathcal{A} + I - \operatorname{diag}\left(X\mathcal{A}\right),\, \mathcal{Y} = Y \odot \mathcal{A} - \operatorname{diag}\left(X\mathcal{A}\right). \end{aligned}$$

Introduce the auxiliary function $\eta : \mathbb{R} \to \mathbb{R}_+$,

$$\eta(t) = \lambda_N \left(\mathbb{E}\left[W(t)^2\right] - J\right). \tag{5.31}$$

To prove that $\phi(C)$ is convex, it suffices to prove that the function ϕ is convex. Introduce $\mathcal{Z}(t)$ and compute successively

$$\begin{aligned} \mathcal{Z}(t) &= W(t)^2 - J \qquad (5.32)\\ &= (\mathcal{X} + t\mathcal{Y})^2 - J \qquad (5.33)\\ &= t^2\, \mathcal{Y}^2 + t\, (\mathcal{X}\mathcal{Y} + \mathcal{Y}\mathcal{X}) + \mathcal{X}^2 - J \qquad (5.34)\\ &= t^2\, \mathcal{Z}_2 + t\, \mathcal{Z}_1 + \mathcal{Z}_0. \qquad (5.35) \end{aligned}$$

The random matrices $\mathcal{Z}_2$, $\mathcal{Z}_1$, and $\mathcal{Z}_0$ do not depend on t. Also, $\mathcal{Z}_2$ is semidefinite positive. The function $\eta(t)$ can be expressed as

$$\eta(t) = \lambda_N \left(\mathbb{E}\left[\mathcal{Z}(t)\right]\right).$$

We will now derive that

$$\mathcal{Z}\left((1-\alpha)t + \alpha u\right) \preceq (1-\alpha)\, \mathcal{Z}(t) + \alpha\, \mathcal{Z}\left(u\right),\ \ \forall \alpha \in [0,1]\,,\ \forall t, u \in \mathbb{R}. \tag{5.36}$$

Since $\eta(t) = t^2$ is convex, the following inequality holds:

$$\left[(1-\alpha)t + \alpha u\right]^2 \leq (1-\alpha)t^2 + \alpha u^2,\ \ \alpha \in [0,1]\,. \tag{5.37}$$

Since the matrix $\mathcal{Z}_2$ is positive semidefinite, (5.37) implies that:

$$\Big(((1-\alpha)t+\alpha u)^2\Big)\,\mathcal{Z}_2 \preceq (1-\alpha)\,t^2\,\mathcal{Z}_2 + \alpha\,u^2\,\mathcal{Z}_2,\ \alpha\in[0,1]$$

After adding to both sides $((1-\alpha)t+\alpha u)\ \mathcal{Z}_1+\mathcal{Z}_0$, we get (5.36). Taking the expectation to both sides of (5.36), get:

$$\begin{aligned}\mathbb{E}\,[\,\mathcal{Z}\,((1-\alpha)t+\alpha u)\,] &\preceq \mathbb{E}\,[\,(1-\alpha)\mathcal{Z}(t)+\alpha\mathcal{Z}\,(u)\,]\\ &= (1-\alpha)\mathbb{E}\,[\,\mathcal{Z}\,(t)\,]+\alpha\mathbb{E}\,[\,\mathcal{Z}\,(u)\,],\ \alpha\in[0,1]\,.\end{aligned}$$

Now, we have that:

$$\begin{aligned}\eta\,((1-\alpha)t+\alpha u) &= \lambda_N\,(\,\mathbb{E}\,[\mathcal{Z}\,((1-\alpha)t+\alpha u)]\,)\\ &\le \lambda_N\,(\,(1-\alpha)\mathbb{E}\,[\mathcal{Z}(t)]+\alpha\mathbb{E}\,[\mathcal{Z}\,(u)]\,)\\ &\le (1-\alpha)\,\lambda_N\,(\,\mathbb{E}\,[\mathcal{Z}(t)]\,)+\alpha\,\lambda_N\,(\,\mathbb{E}\,[\mathcal{Z}\,(u)]\,)\\ &= (1-\alpha)\,\eta(t)+\alpha\,\eta(u),\ \alpha\in[0,1]\,.\end{aligned}$$

The last inequality holds since $\lambda_N(\cdot)$ is convex. This implies $\eta(t)$ is convex and hence $\phi(C)$ is convex.

5.3.4 Fully connected random network: Closed form solution

To get some insight how the optimal weights depend on the network parameters, we consider the impractical, but simple geometry of a complete random symmetric graph. For this example, the optimization problem (5.26) admits a closed form solution, while, in general, numerical optimization is needed to solve (5.26). Although not practical, this example provides insight how the optimal weights depend on the network size N, the link occurrence probabilities, and the link occurrence spatial correlations. The supergraph is symmetric, fully connected, with N nodes and $M=N(N-1)/2$ undirected links. We assume that all the links have the same occurrence probability, i.e., that $\mathbb{P}\,(q_l=1)=\pi_l=p$, $p\in(0,1]$, $l=1,...,M$. We assume that the cross-variance between any pair of links i and j equals to $[R_q]_{ij}=\beta\,p(1-p)$, where β is the correlation coefficient. The matrix R_q is given by

$$R_q = p(1-p)\left[(1-\beta)I+\beta\,11^\top\right].$$

The eigenvalues of R_q are $\lambda_N(R_q) = p(1-p)\,(1+(M-1)\,\beta)$, and $\lambda_i(R_q) = p(1-p)\,(1-\beta) \geq 0$, $i = 2, ..., M$. The condition that $R_q \succeq 0$ implies that $\beta \geq -1/(M-1)$. Also, we have that

$$\beta := \frac{\mathbb{E}\left[q_i q_j\right] - \mathbb{E}\left[q_i\right]\mathbb{E}\left[q_j\right]}{\sqrt{\mathrm{Var}(q_i)}\sqrt{\mathrm{Var}(q_j)}} \tag{5.38}$$

$$= \frac{\mathbb{P}\left(q_i = 1, q_j = 1\right) - p^2}{p(1-p)} \geq -\frac{p}{1-p}. \tag{5.39}$$

Thus, the range of β is restricted to

$$\max\left(\frac{-1}{M-1}, \frac{-p}{1-p}\right) \leq \beta \leq 1. \tag{5.40}$$

Due to the problem symmetry, the optimal weights for all links are the same, say $C^\star$. The expressions for the optimal weight $C^\star$ and for the optimal convergence rate $\phi^\star$ can be obtained after careful manipulations and expressing the matrix $\mathbb{E}\left[W^2\right] - J$ explicitly in terms of p and β; then, it is easy to show that:

$$C^\star = \frac{1}{Np + (1-p)\,(2 + \beta(N-2))} \tag{5.41}$$

$$\phi^\star = 1 - \frac{1}{1 + \frac{1-p}{p}\left(\frac{2}{N}(1-\beta) + \beta\right)}. \tag{5.42}$$

The optimal weight $C^\star$ decreases as β increases. This is intuitive, since positive correlations imply that the links emanating from the same node tend to occur simultaneously, and thus the weight should be smaller. Similarly, negative correlations imply that the links emanating from the same node tend to occur exclusively, which results in larger weights. Finally, we observe that in the uncorrelated case ($\beta = 0$), as N becomes very large, the optimal weight behaves as $1/(Np)$. Thus, for the uncorrelated links and large network, the optimal strategy (at least for this example) is to rescale the supergraph-optimal weight $1/N$ by its occurrence probability p. Finally, for fixed p and N, the fastest rate is achieved when β is as negative as possible.

5.3.5 Numerical optimization: subgradient algorithm

We solve the optimization problem in (5.26) for generic networks by the subgradient algorithm, [92]. In this subsection, we consider spatially uncorrelated links, and we comment on extensions for spatially correlated links. Expressions for spatially correlated links are provided in Appendix C.

We recall that the function $\phi(W)$ is convex (proved in Section 5.3.3). It is nonsmooth because $\lambda_N(\cdot)$ is nonsmooth. Let $H \in \mathbb{S}^N$ be the subgradient of the function $\phi(C)$. To derive the expression for the

subgradient of $\phi(C)$, we use the variational interpretation of $\phi(C)$:

$$\phi(C) \;=\; \max_{v^\top v=1} v^\top \left(\mathbb{E}\left[W^2\right] - J\right) v = \max_{v^\top v=1} f_v(C). \tag{5.43}$$

By the subgradient calculus, a subgradient of $\phi(C)$ at point C is equal to a subgradient H_u of the function $f_u(C)$ for which the maximum of the optimization problem (5.43) is attained, see, e.g., [92]. The maximum of $f_v(C)$ (with respect to v) is attained at $v = u$, where u is the eigenvector of the matrix $\mathbb{E}\left[W^2\right] - J$ that corresponds to its maximal eigenvalue, i.e., the maximal eigenvector. In our case, the function $f_u(C)$ is differentiable (quadratic function), and hence the subgradient of $f_u(C)$ (and also the subgradient of $\phi(C)$) is equal to the gradient of $f_u(C)$, [92]:

$$H_{ij} = \begin{cases} u^\top \frac{\partial\left(\mathbb{E}\left[W^2\right]-J\right)}{\partial C_{ij}} u & \text{if } (i,j) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{5.44}$$

We compute for $(i,j) \in E$

$$\begin{aligned} H_{ij} &= u^\top \frac{\partial\left(\overline{W}^2 - J + R_{\text{corr}}\right)}{\partial C_{ij}} u & (5.45)\\ &= u^\top \left(-2\,\overline{W}\,P_{ij}(e_i-e_j)(e_i-e_j)^\top + 4\,C_{ij}\,P_{ij}(1-P_{ij})(e_i-e_j)(e_i-e_j)^\top\right) u \\ &= 2P_{ij}(u_i-u_j)u^\top(\overline{W}_j - \overline{W}_i) + 4P_{ij}(1-P_{ij})C_{ij}(u_i-u_j)^2 & (5.46) \end{aligned}$$

The subgradient algorithm is given by Algorithm 4. The stepsize α_k is nonnegative, diminishing, and

Algorithm 4 Subgradient algorithm

Set initial $W^{(1)} \in \mathcal{C}$
Set $k = 1$
Repeat
  Compute a subgradient $H^{(k)}$ of ϕ at $W^{(k)}$, and set $W^{(k+1)} = W^{(k)} - \alpha_k H^{(k)}$
  $k := k + 1$

nonsummable: $\lim_{k\to\infty}\alpha_k = 0$, $\sum_{k=1}^{\infty}\alpha_k = \infty$. We choose $\alpha_k = \frac{1}{\sqrt{k}}$, $k = 1, 2, ...$

5.4 Weight optimization: asymmetric random links

We now address the weight optimization for asymmetric random networks. Subsections 5.4.1 and V-B introduce the optimization criterion and the corresponding weight optimization problem, respectively. Subsection V-C shows that this optimization problem is convex.

5.4.1 Optimization criterion: Mean square deviation convergence rate

Introduce now

$$\psi(C) := \lambda_N \left(\mathbb{E}\left[W^\top (I - J) W \right] \right). \tag{5.47}$$

Reference [1] shows that the mean square deviation MSdev satisfies the following equation:

$$\mathrm{MSdev}(k+1) \leq \psi(C)\, \mathrm{MSdev}(k), \ \ \forall k \geq 0. \tag{5.48}$$

Thus, if the quantity $\psi(C)$ is strictly less than one, then MSdev converges to zero asymptotically, with the worst case rate equal to $\psi(C)$. We remark that the condition (5.10) is not needed for (5.48) to hold, i.e., MSdev converges to zero even if condition (5.10) is not satisfied; this condition is needed only for (5.25) to hold, i.e., only to have MSE to converge to zero.

5.4.2 Asymmetric network: Weight optimization problem formulation

In the case of asymmetric links, we optimize the mean square deviation convergence rate, i.e., we solve the following optimization problem:

$$\begin{array}{ll} \text{minimize} & \psi(C) \\ \text{subject to} & W \in \mathcal{C}^{\text{asym}} \\ & \sum_{i=1}^{N} P_{ij} C_{ij} = 1, \ i = 1, ..., N \end{array} \tag{5.49}$$

The constraints in the optimization problem (5.49) assure that, in expectation, condition (5.10) is satisfied, i.e., that

$$1^\top \mathbb{E}\left[W\right] = 1^\top. \tag{5.50}$$

If (5.50) is satisfied, then the consensus algorithm converges to the true average x_{avg} in expectation [1].

Equation (5.50) is a linear constraint with respect to the weights C_{ij}, and thus does not violate the convexity of the optimization problem (5.49). We emphasize that in the case of asymmetric links, we do not assume the weights C_{ij} and C_{ji} to be equal. In section 5.5.2, we show that allowing C_{ij} and C_{ji} to be different leads to better solutions in the case of asymmetric networks.

5.4.3 Convexity of the weight optimization problem

We show that the function $\psi(C)$ is convex. We remark that reference [1] shows that the function is convex, when all the weights C_{ij} are equal to g. We show here that this function is convex even when the weights are different.

Lemma 5.4 (Convexity of $\psi(C)$) The function $\phi : \mathcal{C}^{\text{asym}} \to \mathbb{R}_+$ is convex.

The proof is very similar to the proof of Lemma 5.3. The proof starts with introducing C as in (5.29) and with introducing $W(t)$ as in (5.30). The difference is that, instead of considering the matrix $W^2 - J$, we consider now the matrix $W^\top (I - J) W$. In the proof of Lemma 5.3, we introduced the auxiliary function $\eta(t)$ given by (5.31); here, we introduce the auxiliary function $\kappa(t)$, given by:

$$\kappa(t) = \lambda_N \left(W(t)^\top (I - J) W(t) \right), \tag{5.51}$$

and show that $\psi(C)$ is convex by proving that $\kappa(t)$ is convex. Then, we proceed as in the proof of Lemma 5.3. In (5.35) the matrix $\mathcal{Z}_2$ becomes $\mathcal{Z}_2 := \mathcal{Y}^\top (I - J) \mathcal{Y}$. The random matrix $\mathcal{Z}_2$ is obviously positive semidefinite. The proof then proceeds as in Lemma 5.3.

5.5 Simulations

We demonstrate the effectiveness of our approach with a comprehensive set of simulations. These simulations cover both examples of asymmetric and symmetric networks and both networks with random link failures and with randomized protocols. In particular, we consider the following two standard sets of experiments with random networks: 1) spatially correlated link failures and symmetric links and 2) randomized protocols, in particular, the broadcast gossip algorithm [1]. With respect to the first set, we consider correlated link failures with two types of correlation structure. We are particularly interested in studying the dependence of the performance and of the gains on the size of the network N and on the link correlation structure.

In all these experiments, we consider geometric random graphs. Nodes communicate among themselves if within their radius of communication, r. The nodes are uniformly distributed on a unit square. The number of nodes is $N = 100$ and the average degree is $15\%N$ (= average number of neighbors across all nodes.) In subsection 5.5.1, the random instantiations of the networks are undirected; in subsection VI-B, the random instantiations of the networks are directed.

In the first set of experiments with correlated link failures, the link occurrence probabilities P_{ij} are chosen such that they decay quadratically with the distance:

$$P_{ij} = 1 - k \left(\frac{\delta_{ij}}{r} \right)^2, \tag{5.52}$$

where we choose $k = 0.7$. We see that, with (5.52), a link will be active with high probability if the nodes are close ($\delta_{ij} \simeq 0$), while the link will be down with probability at most 0.7, if the nodes are apart by r.

We recall that we refer to our weight design, i.e., to the solutions of the weight optimization problems (5.26), (5.49), as probability based weights (PBW). We study the performance of PBW, comparing it with the standard weight choices available in the literature: in subsection 5.5.1, we compare it with the Metropolis weights (MW), discussed in [91], and the supergraph based weights (SGBW). The SGBW are the optimal (nonnegative) weights designed for a static (nonrandom) graph G, which are then applied to a random network when the underlying supergraph is G. This is the strategy used in [85]. For asymmetric links (and for asymmetric weights $C_{ij} \neq C_{ji}$), in subsection 5.5.2, we compare PBW with the optimal weight choice in [1] for broadcast gossip that considers all the weights to be equal.

In the first set of experiments in subsection 5.5.1, we quantify the performance gain of PBW over SGBW and MW by the gains:

$$\Gamma_s^\tau = \frac{\tau_{\mathrm{SGBW}}}{\tau_{\mathrm{PBW}}} \tag{5.53}$$

where τ is a time constant defined as:

$$\tau = \frac{1}{0.5 \ln \phi(C)} \tag{5.54}$$

We also compare PBW with SGBW and MW with the following measure:

$$\Gamma_s^\eta = \frac{\eta_{\mathrm{SGBW}}}{\eta_{\mathrm{PBW}}} \tag{5.55}$$

$$\Gamma_m^\eta = \frac{\eta_{\mathrm{MW}}}{\eta_{\mathrm{PBW}}} \tag{5.56}$$

where η is the asymptotic time constant defined by

$$\eta = \frac{1}{|\gamma|} \tag{5.57}$$

$$\gamma = \lim_{k \to \infty} \left(\frac{\|e(k)\|}{\|e(0)\|} \right)^{1/k} \tag{5.58}$$

For random networks η is an almost sure constant and τ is an upper bound on η, [85].

Subsections 5.5.1 and 5.5.2 will provide further details on the expermints.

5.5.1 Symmetric links: random networks with correlated link failures

To completely define the probability distribution of the random link vector $q \in \mathbb{R}^M$, we must assign probability to each of the 2^M possible realizations of q, $q = (\alpha_1, ..., \alpha_M)^\top$, $\alpha_i \in \{0, 1\}$. Since in networks of practical interest M may be very large, of order 1000 or larger, specifying the complete distribution of the vector q is most likely infeasible. Hence, we work with the second moment description and specify only the first two moments of its distribution, the mean and the covariance, π and R_q. Without loss of generality, order the links so that $\pi_1 \leq \pi_2 \leq ... \leq \pi_M$.

Lemma 5.5 The mean and the variance (π, R_q) of a Bernoulli random vector satisfy:

$$0 \;\leq\; \pi_i \;\leq\; 1,\; i = 1, ..., N \tag{5.59}$$

$$R_q \;\succeq\; 0 \tag{5.60}$$

$$\max\left(-\pi_i\pi_j,\, \pi_i + \pi_j - 1 - \pi_i\,\pi_j\right) \;\leq\; [R_q]_{ij} \leq \pi_i\,(1 - \pi_j) = \overline{R}_{ij},\; i < j \tag{5.61}$$

Equations (5.59) and (5.60) must hold because π_l's are probabilities and R_q is a covariance matrix. Recall that

$$[R_q]_{ij} = \mathbb{E}\,[q_i q_j] - \mathbb{E}\,[q_i]\,\mathbb{E}\,[q_j] = \mathbb{P}\,(q_i = 1,\, q_j = 1) - \pi_i\pi_j. \tag{5.62}$$

To prove the lower bound in (5.61), observe that:

$$\begin{aligned}\mathbb{P}\,(q_i = 1,\, q_j = 1) \;&=\; \mathbb{P}\,(q_i = 1) + \mathbb{P}\,(q_j = 1) - \mathbb{P}\,(\{q_i = 1\} \text{ or } \{q_j = 1\}) \\ &=\; \pi_i + \pi_j - \mathbb{P}\,(\{q_i = 1\} \text{ or } \{q_j = 1\}) \geq \pi_i + \pi_j - 1.\end{aligned} \tag{5.63}$$

In view of the fact that $\mathbb{P}\,(q_i = 1,\, q_j = 1) \geq 0$, (5.63), and (5.62), the proof for the lower bound in (5.61) follows. The upper bound in (5.61) holds because $\mathbb{P}\,(q_i = 1,\, q_j = 1) \leq \pi_i,\; i < j$ and (5.62).

If we choose a pair (π, R_q) that satisfies (5.59), (5.60), (5.61), one cannot guarantee that (π, R_q) is a valid pair, in the sense that there exists a probability distribution on q with its first and second moments being equal to (π, R_q), [93]. Furthermore, if (π, R_q) is given, to simulate binary random variables with the marginal probabilities and correlations equal to (π, R_q) is challenging. These questions have been studied, see [94, 93]. We use the results in [94, 93] to generate our correlation models. In particular, we use the result that $\overline{R} = \left[\overline{R}_{ij}\right]$ (see (5.61)) is a valid correlation structure for any π, [94]. We simulate the correlated

links by the method proposed in [93]; this method handles a wide range of different correlation structures and has a small computational cost.

Link correlation structures. We consider two different correlation structures for any pair of links i and j in the supergraph:

$$[R_q]_{ij} = c_1 \, \overline{R}_{ij} \tag{5.64}$$

$$[R_q]_{ij} = c_2 \, \theta^{\kappa_{ij}} \, \overline{R}_{ij} \tag{5.65}$$

where $c_1 \in (0, 1]$, $\theta \in (0, 1)$ and $c_2 \in (0, 1]$ are parameters, and κ_{ij} is the distance between links i and j defined as the length of the shortest path that connects them in the supergraph.

The correlation structure (5.64) assumes that the correlation between any pair of links is a fraction of the maximal possible correlation, for the given π (see (5.61) to recall $\overline{R}_{ij}$). Reference [94] constructs a method for generating the correlation structure (5.64).

The correlation structure (5.65) assumes that the correlation between the links decays geometrically with this distance . In our simulations, we set $\theta = 0.95$, and find the maximal c_2, such that the resulting correlation structure can be simulated by the method in [93]. For all the networks that we simulated in the chapter, c_2 is between 0.09 and 0.11.

Results. We want to address the following two questions: 1) What is the performance gain (Γ_s, Γ_m of PBW over SGBW and MW; and 2) How does this gain scale with the network size, i.e., the number of nodes N?

Performance gain of PBW over SGBW and MW. We consider question 1) for both correlation structures (5.64), (5.65). We generate 20 instantiations of our standard supergraphs (with 100 nodes each and approximately the same average relative degree, equal to 15%). Then, for each supergraph, we generate occurrence probabilities according to rule (5.52). For each supergraph with the given occurrence probabilities, we generate two link correlation structures, (5.64) and (5.65). We evaluate the convergence rate ϕ_j given by (5.25), time constants η_j given by (5.57), and τ_j, given by (5.54), and the performance gains $[\Gamma_s^\eta]_j$, $[\Gamma_m^\eta]_j$ for each supergraph ($j = 1, ..., 20$). We compute the mean $\overline{\phi}$, the maximum ϕ^+ and the minimum ϕ^- from the list $\{\phi_j\}$, $j = 1, ..., 20$ (and similarly for $\{\eta_j\}$ and $\{\tau_j\}$, $j = 1, ..., 20$). Results for the correlation structure (5.64) are given in Table 5.1 and for the correlation structure (5.65), in Table 5.2. The performance gains Γ_s, Γ_m, for both correlation structures are in Table 5.3. In addition, Figure 5.1 depicts the averaged error norm over 100 sample paths. We can see that the PBW show better performance than the SGBW and the MW for both correlation structures (5.64) and (5.65). For example, for the correlation (5.64), the PBW take less than 40 iterations to achieve 0.2% precision, while the SGBW take more than 70, and the MW take

Table 5.1: Correlation structure (5.64): Average $\overline{(\cdot)}$, maximal $(\cdot)^+$, and minimal $(\cdot)^-$ values of the MSE convergence rate ϕ (5.23), and corresponding time constants τ (5.54) and η (5.57), for 20 generated supergraphs

	SGBW	PBW	MW
$\overline{\phi}$	0.91	0.87	
ϕ^+	0.95	0.92	
ϕ^-	0.89	0.83	
$\overline{\tau}$	22.7	15.4	
τ^+	28	19	
τ^-	20	14	
$\overline{\eta}$	20	13	29
η^+	25	16	38
η^-	19	12	27

more than 80 iterations. For correlation (5.65), to achieve 0.2% precision, the PBW take about 47 iterations, while the SGBW and the MW take more than 90 and 100 iterations, respectively. The average performance

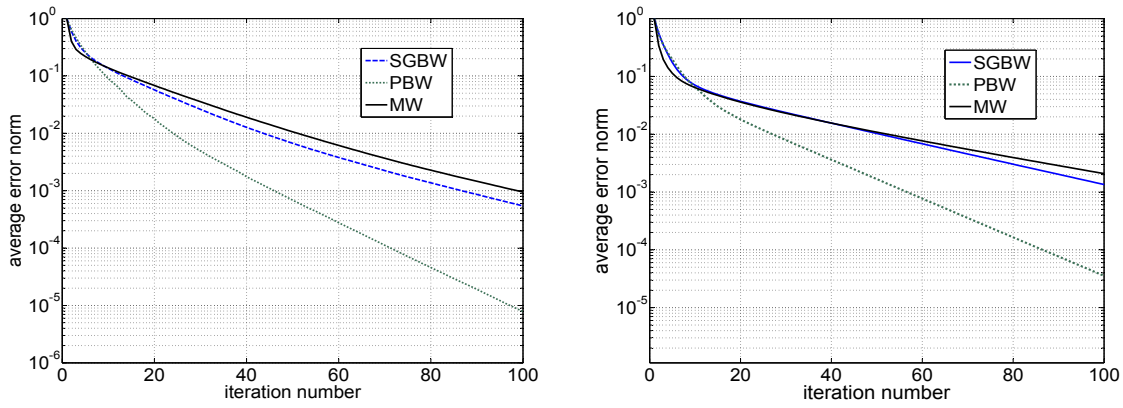

Figure 5.1: Average error norm versus iteration number. Left: correlation structure (5.64); right: correlation structure (5.65).

gain of PBW over MW is larger than the performance gain over SGBW, for both (5.64) and (5.65). The gain over SGBW, Γ_s, is significant, being 1.54 for (5.64) and 1.73 for (5.65). The gain with the correlation structure (5.65) is larger than the gain with (5.64), suggesting that larger gain over SGBW is achieved with smaller correlations. This is intuitive, since large positive correlations imply that the random links tend to occur simultaneously, i.e., in a certain sense random network realizations are more similar to the underlying supergraph.

Notice that the networks with R_q as in (5.65) achieve faster rate than for (5.64) (having at the same time similar supergraphs and occurrence probabilities). This is in accordance with the analytical studies in section 5.3.4 that suggest that faster rates can be achieved for smaller (or negative correlations) if G and π are fixed.

Table 5.2: Correlation structure (5.65): Average $\overline{(\cdot)}$, maximal $(\cdot)^+$, and minimal $(\cdot)^-$ values of the MSE convergence rate ϕ (5.23), and corresponding time constants τ (5.54) and η (5.57), for 20 generated supergraphs

	SGBW	PBW	MW
$\overline{\phi}$	0.92	0.86	
ϕ^+	0.94	0.90	
ϕ^-	0.91	0.84	
$\overline{\tau}$	25.5	14.3	
τ^+	34	19	
τ^-	21	12	
$\overline{\eta}$	20	11.5	24.4
η^+	23	14	29
η^-	16	9	19

Table 5.3: Average $\overline{(\cdot)}$, maximal $(\cdot)^+$, and minimal $(\cdot)^-$ performance gains Γ_s^η and Γ_m^η (5.55) for the two correlation structures (5.64) and (5.65) for 20 generated supergraphs

	Correlation (5.64)	Correlation (5.65)
$\overline{(\Gamma_s^\eta)}$	1.54	1.73
$(\Gamma_s^\eta)^+$	1.66	1.91
$(\Gamma_s^\eta)^-$	1.46	1.58
$\overline{(\Gamma_m^{eta})}$	2.22	2.11
$(\Gamma_m^\eta)^+$	2.42	2.45
$(\Gamma_m^\eta)^-$	2.07	1.92

Performance gain of PBW over SGBW as a function of the network size. To answer question 2), we generate the supergraphs with N ranging from 30 up to 160, keeping the average relative degree of the supergraph approximately the same (15%). Again, PBW performs better than MW ($\tau_{\mathrm{SGBW}} < 0.85\tau_{\mathrm{MW}}$), so we focus on the dependence of Γ_s on N, since it is more critical.

Figure 5.2 plots Γ_s versus N, for the two correlation structures. The gain Γ_s increases with N for both (5.65) and (5.64).

5.5.2 Broadcast gossip algorithm [1]: Asymmetric random links

In the previous section, we demonstrated the effectiveness of our approach in networks with random symmetric link failures. This section demonstrates the validity of our approach in randomized protocols with asymmetric links. We study the broadcast gossip algorithm [1]. Although the optimization problem (5.49) is convex for generic spatially correlated directed random links, we pursue here numerical optimization of the broadcast gossip algorithm proposed in [1], where, at each time step, node i is selected at random, with probability $1/N$. Node i then broadcasts its state to all its neighbors within its wireless range. The neighbors then update their state by performing the weighted average of the received state with their own state. The nodes outside the set Ω_i and the node i itself keep their previous state unchanged. The broadcast gossip

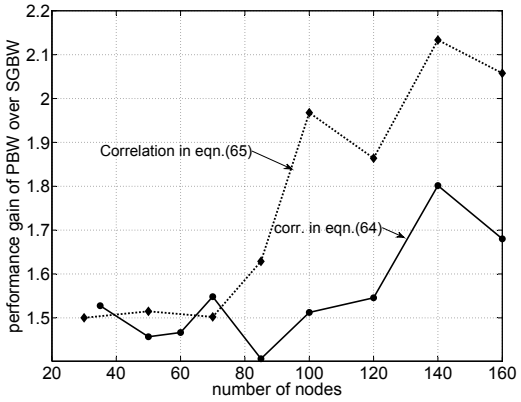

Figure 5.2: Performance gain of PBW over SGBW (Γ_s^{η}, (5.55)) as a function of the number of nodes in the network.

algorithm is well suited for WSN applications, since it exploits the broadcast nature of wireless media and avoids bidirectional communication [1].

Reference [1] shows that, in broadcast gossiping, all the nodes converge a.s. to a common random value c with mean x_{avg} and bounded mean squared error. Reference [1] studies the case when the weights $C_{ij} = g$, $\forall (i,j) \in E$ and finds the optimal $g = g^{\star}$ that optimizes the mean square deviation MSdev (see (5.49)). We optimize the same objective function (see (5.49)) as in [1], but allowing different weights for different directed links. We detail on the numerical optimization for the broadcast gossip in the Appendix C. We consider again the supergraph G from our standard experiment with $N = 100$ and average degree $15\%N$. For the broadcast gossip, we compare the performance of PBW with 1) the optimal equal weights in [1] with $C_{ij} = g^{\star}$, $(i,j) \in E$; 2) broadcast gossip with $C_{ij} = 0.5$, $(i,j) \in E$.

Figure 5.3 (left) plots the consensus mean square deviation MSdev for the 3 different weight choices. The decay of MSdev is much faster for the PBW than for $C_{ij} = 0.5$, $\forall\,(i,j)$ and $C_{ij} = g^{\star}$, $\forall\,(i,j)$. For example, the MSdev falls below 10% after 260 iterations for PBW (i.e., 260 broadcast transmissions); broadcast gossip with $C_{ij} = g^{\star}$ and $C_{ij} = 0.5$ take 420 transmissions to achieve the same precision. This is to be expected, since PBW has many moredegrees of freedom for to optimize than the broadcast gossip in [1] with all equal weights $C_{ij} = g^{\star}$. Figure 5.3 (right) plots the MSE, i.e., the deviation of the true average x_{avg}, for the three weight choices. PBW shows faster decay of MSE than the broadcast gossip with $C_{ij} = g^{\star}$ and $C_{ij} = 0.5$. The weights provided by PBW are different among themselves, varying from 0.3 to 0.95. The weights C_{ij} and C_{ji} are also different, where the maximal difference between C_{ij} and C_{ji}, $(i,j) \in E$, is 0.6. Thus, in the case of directed random networks, asymmetric matrix C results in faster convergence rate.

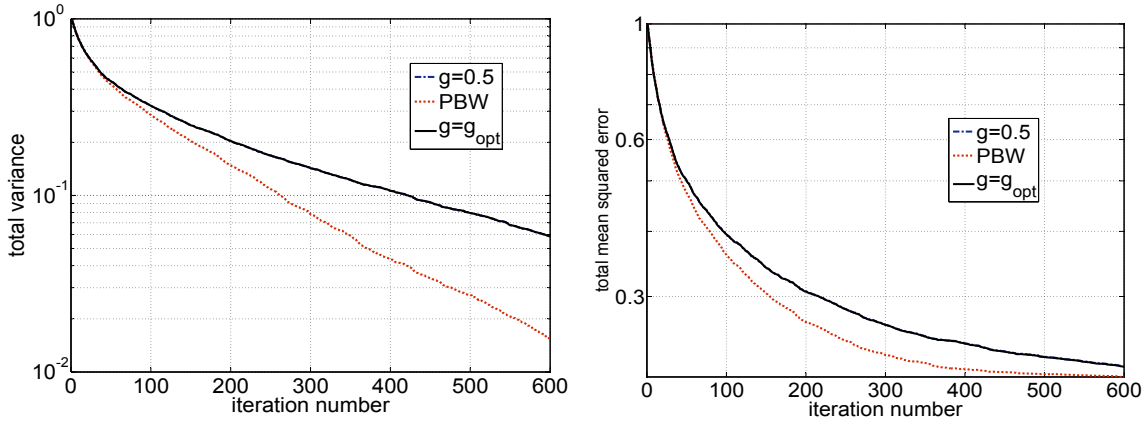

Figure 5.3: Broadcast gossip algorithm with different weight choices. Left: total variance; right: total mean squared error.

5.5.3 Distributed optimization of Huber losses via D–NG algorithm

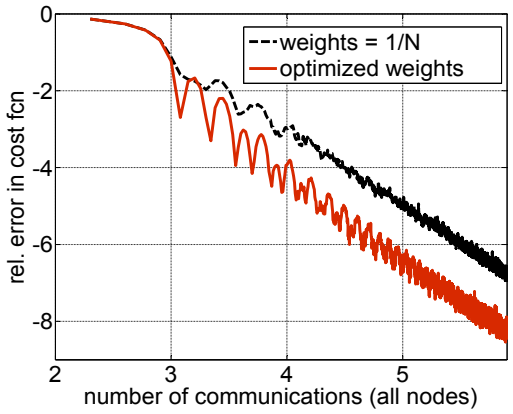

Figure 5.4: Average normalized optimality gap $\mathbf{err_f}$ versus $\mathcal{K}'$ (in the $\log_{10} - \log_{10}$ scale) for the $N = 20$-node network and mD–NG method with different weight assignments; Red, solid line: optimized, PBW weights; black, dotted line: $C_{ij} = 1/N, \forall\{i,j\} \in E$.

We have seen in Chapter 3 that the convergence constant of our distributed optimization algorithms, e.g., mD–NG, depends on the mean square convergence rate, $\phi(C) = \overline{\mu}^2$ through the inverse spectral gap $\frac{1}{1-\overline{\mu}}$. We provide here a simulation example to show that PBW significantly reduce the number of communications and gradient evaluations to achieve a certain accuracy.

We simulate mD–NG on a $N = 20$-node random network. We compare the mD–NG's performance with: 1) the uniform weights $C_{ij} = 1/N$; and 2) the PBW weights. Clearly, the optimized weights correspond to a smaller (better) parameter $\overline{\mu}$. The supergraph $\mathcal{G}$ has $N = 20$ nodes and 91 links; it is generated as a geometric graph (as in the previous simulation), with a connectivity radius $\delta_0 = 0.55$. (Only the nodes

whose distance is less or equal δ_0 are connected by an edge.) The links $\{i,j\} \in E$ are spatio-temporally independent, and the link failure probabilities are generated as: $P_{ij} = 0.5 \times \frac{\delta_{ij}^2}{\delta_0^2}$. We consider Huber losses f_i's, with: 1) for $i = 1, 2, ..., 7$, $\theta_i = \theta^\bullet(1 + \nu_i)$; and 2) for $j = 8, 9, ..., 20$, $\theta_j = (-\theta^\bullet)(1 + \nu_j)$. Here $\theta^\bullet = 4$ and ν_i's and ν_j's are generated randomly from the uniform distribution on $[-0.1, 0.1]$. We generate one random simulation run and compare the two weight choices. Figure 4.4 plots $\mathbf{err_f}$ versus $\mathcal{K}'$ for the two weight choices. First, we can see that the weight optimization does not affect the convergence rate (the slopes of the two lines match), but it improves the convergence constant. This is in accordance with the established upper bounds in Theorem 3.8. Second, the optimized weights significantly reduce the communication cost for higher accuracies. For example, for the precision 10^{-4}, the method with the optimized weights requires about 1.12×10^4 transmissions, while the method with uniform weights requires about 3.15×10^4 transmissions for the same accuracy.

5.6 Conclusion

In this chapter, we studied the optimization of the weights for the consensus algorithm under random topology and spatially correlated links. We consider both networks with random link failures and randomized algorithms; from the weights optimization point of view, both fit into the same framework. We show that, for symmetric random links, optimizing the MSE convergence rate is a convex optimization problem, and for asymmetric links, optimizing the mean squared deviation from the current average state is also a convex optimization problem. We illustrate with simulations the performance of our probability based weights (PBW) and compare them with existing weight assignments, both for distributed averaging and distributed optimization problems.

Chapter 6

Distributed Augmented Lagrangian Methods: General Problems

6.1 Introduction

In this Chapter, we develop distributed augmented Lagrangian (AL) algorithms of type (1.6)–(1.7) that utilize asynchronous communication. We assume a very general distributed optimization model

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{N} f_i(x) \\ \text{subject to} & x \in \mathcal{X}_i,\ i = 1, ..., N \end{array} . \tag{6.1}$$

Here N is the number of nodes in the network, the private cost functions $f_i : \mathbb{R}^d \to \mathbb{R}$ are convex, and each $f_i(\cdot)$ is known locally only by node i. The sets $\mathcal{X}_i$ are *private*, closed, convex constraint sets. We remark that (6.1) captures the scenario when, in addition to private constraints, there is a public constraint $x \in X$ (where X is a closed, convex set,) by replacing $\mathcal{X}_i$ with $\mathcal{X}_i \cap X$.

This chapter proposes a novel augmented Lagrangian (AL) dual distributed algorithm for solving (6.1), which handles private costs $f_i(\cdot)$, private constraints $\mathcal{X}_i$, and is resilient to random communication failures. We refer to this algorithm as AL–G (augmented Lagrangian gossiping.) We also consider two variants to AL–G, namely, the AL–MG (augmented Lagrangian multiple neighbor gossiping) and the AL–BG (augmented Lagrangian broadcast gossiping.) The AL–G and AL–MG algorithms use *unidirectional* gossip communication (see, e.g., [10]). For networks with reliable communication (i.e., no failures,) we propose the simplified AL–BG algorithm with reduced communication, reduced computation, and lower data storage cost. Our algorithms update the dual variables by the standard method of multipliers, [15], synchronously,

at a slow time scale, and update the primal variables with a *novel,* Gauss-Seidel type (see, e.g., [37]) randomized algorithm with asynchronous gossip communication, at a fast time scale. Proof of convergence for the method of multipliers (for the dual variables update) is available in the literature, e.g., [15]. However, our algorithms to update primal variables (referred to as P–AL–G (primal AL gossip), P–AL–MG and P–AL–BG) are novel; a major contribution of this chapter is to prove convergence of the P–AL–G, for private constraints, under very generic network topologies, random link failures, and gossip communication. The proof is then adapted to P–AL–MG and P–AL–BG.

We provide two simulation examples, namely, l_1–regularized logistic regression for classification and cooperative spectrum sensing for cognitive radio networks. These simulation examples: 1) corroborate convergence of the proposed algorithms; and 2) compare their performance, in terms of communication and computational cost, with the algorithms in [24, 22, 8, 5].

As far as we are aware, distributed AL dual algorithms have been studied only for *static networks*. For example, references [8, 5] consider a special case of (6.1), namely, the Lasso (least-absolute shrinkage and selection operator) type problem. They propose the ADMM (alternating direction method of multipliers) type dual algorithms for *static networks*, *synchronous communication*, and *no constraints.* Reference [78] applies ADMM to various statistical learning problems, including Lasso, support vector machines, and sparse logistic regression, assuming a parallel network architecture (all nodes communicate with a fusion node,) synchronous communication, and no link failures.

In this chapter, we develop a AL dual algorithm for the optimization (6.1) with private costs and private constraints, random networks, and asynchronous gossip communication. In contrast with existing work on dual methods, for example, [8, 5], our AL–G handles *private constraints*, *random networks*, *asymmetric link failures*, and *gossip communication.*[1]

This Chapter does not study convergence rates. We study convergence rates in Chapter 7 for more structured cost functions.

Chapter organization. Section 6.2 introduces the communication and computational model. Section 6.3 presents the AL–G algorithm for the networks with link failures. Section 6.3 analyzes the convergence of the AL–G algorithm. Section 6.4 studies the variants to AL–G, the AL–MG, and AL–BG algorithms. Section 6.6 provides two simulation examples: 1) l_1–regularized logistic regression for classification; and 2) cooperative spectrum sensing for cognitive radios. Finally, section 6.7 concludes the chapter. Appendix D analyzes convergence of AL–MG and AL–BG.

[1] AL–G algorithm uses asynchronous gossip communication, but it is not completely asynchronous algorithm, as it updates the dual variables synchronously, at a slow time scale (as detailed in Section 6.4.)

The results in this chapter have been published in [38].

6.2 Problem model

This section explains the communication model (the time slotting, the communication protocol, and the link failures,) and the computation model (assumptions underlying the optimization problem (6.1).)

Network model: Supergraph

The connectivity of the networked system is described by the bidirectional, connected supergraph $G = (\mathcal{N}, E)$, where $\mathcal{N}$ is the set of nodes (with cardinality $|\mathcal{N}| = N$) and E is the set of bidirectional edges $\{i, j\}$ ($|E| = M$). The supergraph G is simple, i.e., there are no self-edges. Denote by $\Omega_i \subset \mathcal{N}$, the neighborhood set of node i in G, with cardinality $d_i = |\Omega_i|$. The integer d_i is the (supergraph) degree of node i. The supergraph G models and collects all (possibly unreliable) communication channels in the network; actual network realizations during the algorithm run will be *directed* subgraphs of G. We denote the directed edge (arc) that originates in node i and ends in node j either by (i, j) or $i \to j$, as appropriate. The set of all arcs is: $E_d = \{(i, j) : \{i, j\} \in E\}$, where $|E_d| = 2M$. We assume that the supergraph is known, i.e., each node knows a priori with whom it can communicate (over a possibly unreliable link.)

Optimization model

We summarize the assumptions on the cost functions $f_i(\cdot)$ and $f(\cdot)$, $f(x) := \sum_{i=1}^{N} f_i(x)$, and the constraint sets $\mathcal{X}_i$ in (6.1):

Assumption 6.1 We assume the following for the optimization problem (6.1):

1. The functions $f_i : \mathbb{R}^d \to \mathbb{R}$ are convex and coercive, i.e., $f_i(x) \to \infty$ whenever $\|x\| \to \infty$.
2. The constraint sets $\mathcal{X}_i \subset \mathbb{R}^d$ are closed and convex, and $\mathcal{X} := \cap_{i=1}^{N} \mathcal{X}_i$ is nonempty.
3. **(Regularity condition)** There exists a point $x_0 \in \mathrm{ri}\,(\mathcal{X}_i)$, for all $i = 1, ..., N$.

Here $\mathrm{ri}\,(\mathcal{S})$ denotes the relative interior of a set $\mathcal{S} \subset \mathbb{R}^d$.[2]

We will derive the AL–G algorithm to solve (6.1) by first reformulating it (see ahead (6.2),) and then dualizing the reformulated problem (using AL dual.) Assumption 1.3 will play a role to assure strong duality. This will be detailed in subsection III-A. Note that Assumption 1.3 is rather mild, saying only that

[2]Relative interior of a nonempty set $\mathcal{S} \subset \mathbb{R}^d$ is the set: $\{\, x \in \mathcal{S} : \forall y \in \mathcal{S},\, \exists a > 1 :\, ax + (1-a)y \in \mathcal{S} \,\}$.

the intersection of the $\mathcal{X}_i$'s, $i = 1, ..., N$, is "large" enough to contain a point from the relative interior of each of the $\mathcal{X}_i$'s. Denote by $f^\star$ the optimal value and $\mathcal{X}^\star = \left\{x^\star \in \mathcal{X} : \sum_{i=1}^N f_i(x^\star) = f^\star\right\}$ the solution set to (6.1). Under Assumptions 6.1, $f^\star$ is finite, and $\mathcal{X}^\star$ is nonempty, compact, and convex, [81]. The model (6.1) applies also when $\mathcal{X}_i = \mathbb{R}^d$, for i's in a subset of $\{1, ..., N\}$. The functions $f_i(\cdot)$, $f(\cdot)$ need not be differentiable; $f(\cdot)$ satisfies an additional assumption detailed in Section 6.4.

We now reformulate (6.1) to derive the AL–G algorithm. Start by cloning the variable $x \in \mathbb{R}^d$ and attaching a local copy of it, $x_i \in \mathbb{R}^d$, to each node in the network. In addition, introduce the variables $y_{ij} \in \mathbb{R}^d$ and $y_{ji} \in \mathbb{R}^d$, attached to each link $\{i, j\}$ in the supergraph. To keep the reformulated problem equivalent to (6.1), we introduce coupling constraints $x_i = y_{ij},\ (i, j) \in E_d$ and $y_{ij} = y_{ji},\ \{i, j\} \in E$. The reformulated optimization problem becomes:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^N f_i(x_i) \\ \text{subject to} & x_i \in \mathcal{X}_i,\ i = 1, ..., N, \\ & x_i = y_{ij},\ (i, j) \in E_d \\ & y_{ij} = y_{ji},\ \{i, j\} \in E. \end{array} \tag{6.2}$$

The variables x_i and y_{ij} may be interpreted as virtual nodes in the network (see Figure 6.1.) Physically, the

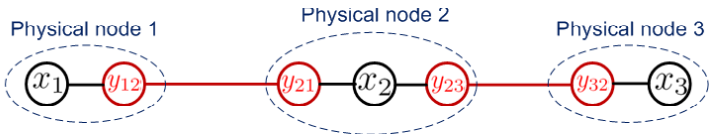

Figure 6.1: Illustration of the reformulation (6.2) for a chain supergraph with $N = 3$ (physical) nodes.

variables x_i, y_{ij}, $j \in \Omega_i$ are maintained by (physical) node i. The virtual link between nodes x_i and y_{ij} is reliable (non-failing,) as both x_i and y_{ij} are physically maintained by node i. On the other hand, the virtual link between y_{ij} and y_{ji} may be unreliable (failing,) as this link corresponds to the physical link between nodes i and j.

The optimization problems (6.1) and (6.2) are equivalent because the supergraph is connected. The optimal value for (6.2) is equal to the optimal value for (6.1) and equals $f^\star$; the set of solutions to (6.2) is $\left\{\{x_i^\star\}, \{y_{ij}^\star\} : x_i^\star = x^\star, \forall i = 1, ..., N,\ y_{ij}^\star = x^\star,\ \forall (i, j) \in E_d, \text{ for some } x^\star \in \mathcal{X}^\star\right\}$.

Time slotting

As we will see in section 6.3., the AL–G algorithm (and also its variants AL–MG and AL–BG in Section 6.5) is based on the AL dual of (6.2). The AL–G operates at 2 time scales: the dual variables are updated at a slow

time scale, and the primal variables are updated at a fast time scale. Thus, accordingly, the time is slotted with: 1) slow time scale slots $\{t\}$; and 2) fast time scale slots $\{k\}$. Fast time scale slots (for the primal variables update) involve asynchronous communication between the nodes in the network and are detailed in the next paragraph. At the end of each t-slot, there is an idle time interval with no communication, when the dual variables are updated. The dual variables update at each node requires no communication.

Fast time scale slots $\{k\}$ and asynchronous communication model

We now define the fast time scale slots $\{k\}$ for the asynchronous communication and the primal variables update. We assume the standard model for asynchronous communication [10]. Each node (both physical and virtual) has a clock that ticks (independently across nodes) according to a λ-rate Poisson process. Denote the clocks of x_i and y_{ij} by T_i^x and T_{ij}^y, respectively. If T_i^x ticks, a virtual communication from y_{ij}, $\forall j \in \Omega_i$, to x_i, follows. With the AL–G algorithm, this will physically correspond to the update of the variable x_i, as we will see later. If the clock T_{ij}^y ticks, then (virtual) node y_{ij} transmits to y_{ji} (physically, node i transmits to node j.) We will see later that, after a (successful) communication $y_{ij} \to y_{ji}$, the update of y_{ji} follows. We also introduce a virtual clock T that ticks whenever one of the clocks T_i^x, T_{ij}^y, ticks; the clock T ticks according to a $(N + 2M)$–rate Poisson process. Denote by τ_k, $k = 1, 2, ...$ the times when the k-th tick of T occurs. The time is slotted and the k-th slot is $[\tau_{k-1}, \tau_k)$, $\tau_0 = 0$, $k = 1, 2, ...$[3]

Random link failures

Motivated by applications in wireless networked systems, we allow that transmissions $y_{ij} \to y_{ji}$ may fail. (Of course, the transmissions through the virtual links $y_{ij} \to x_i$ do not fail.) To formally account for link failures, we define the $N \times N$ random adjacency matrices $A(k)$, $k = 1, 2, ...$; the matrix $A(k)$ defines the set of available physical links at time slot k. We assume that the link failures are temporally independent, i.e., $\{A(k)\}$ are independent identically distributed (i.i.d.) The entries $A_{ij}(k)$, $(i, j) \in E_d$, are Bernoulli random variables, $A_{ij}(k) \sim \text{Bernoulli}(\pi_{ij})$, $\pi_{ij} = \mathbb{P}(A_{ij}(k) = 1) > 0$, and $A_{ij}(k) \equiv 0$, for $(i, j) \notin E_d$. We allow $A_{ij}(k)$ and $A_{lm}(k)$ to be correlated.[4] At time slot k, at most one link $(i, j) \in E_d$ is activated for transmission. If it is available at time k, i.e., if $A_{ij}(k) = 1$, then the transmission is successful; if the link (i, j) is unavailable ($A_{ij}(k) = 0$,) then the transmission is unsuccessful. We assume naturally that the Poisson process that governs the ticks of T and the adjacency matrices $A(k)$, $k = 1, 2, ...$ are independent. Introduce the ordering of links $(i, j) \in E_d$, by attaching a distinct number l, $l = 1, ..., 2M$, to each link

[3]For notation simplicity, at the beginning of each t–slot, we reset τ_0 to zero, and we start counting the k–slots from $k = 1$.
[4]With AL–MG algorithm, in Section 6.6, we will additionally require $A_{ij}(k)$ and $A_{lm}(k)$ be independent.

(i,j); symbolically, we write this as $l \sim (i,j)$. Introduce now the random variables $\zeta(k)$, $k = 1, 2, ...$, defined as follows: 1) $\zeta(k) = i$, if the k-th tick of T comes from T_i^x; 2) $\zeta(k) = N + l$, $l \sim (i,j)$, if the k-th tick of T comes from T_{ij}^y and $A_{ij}(k) = 1$; and 3) $\zeta(k) = 0$, otherwise. It can be shown that $\zeta(k)$, $k = 1, 2, ...$, are i.i.d. The random variables $\zeta(k)$ define the order of events in our communication model. For example, $\zeta = N + l$, $l \sim (i,j)$, means that, at time slot k =1, the virtual node y_{ij} successfully transmitted data to the virtual node y_{ji}. We remark that $\mathbb{P}(\zeta(k) = s)$ is strictly positive, $\forall s = 0, 1, ..., N + 2M$. This fact will be important when studying the convergence of AL–G.

6.3 AL–G algorithm (augmented Lagrangian gossiping)

This section details the AL–G algorithm for solving (6.1). In subsection 6.3.1, we dualize (6.2) to form the AL dual of problem (6.2). Subsection IV-B details the D–AL–G algorithm for the dual variable update, at a slow time scale; subsection IV-C details P–AL–G to update the primal variables, at a fast time scale.

6.3.1 Dualization

We form the AL dual of the optimization problem (6.2) by dualizing all the constraints of the type $x_i = y_{ij}$ and $y_{ij} = y_{ji}$. The dual variable that corresponds to the constraint $x_i = y_{ij}$ will be denoted by $\eta_{(i,j)}$, the dual variable that corresponds to the (different) constraint $x_j = y_{ji}$ will be denoted by $\eta_{(j,i)}$, and the one that corresponds to $y_{ij} = y_{ji}$ is denoted by $\lambda_{\{i,j\}}$. In the algorithm implementation, both nodes i and j will maintain their own copy of the variable $\lambda_{\{i,j\}}$–the variable $\lambda_{(i,j)}$ at node i and the variable $\lambda_{(j,i)}$ at node j. Formally, we use both $\lambda_{(i,j)}$ and $\lambda_{(j,i)}$, and we add the constraint $\lambda_{(i,j)} = \lambda_{(j,i)}$. The term after dualizing $y_{ij} = y_{ji}$, equal to $\lambda_{\{i,j\}}^\top (y_{ij} - y_{ji})$, becomes: $\lambda_{(i,j)}^\top y_{ij} - \lambda_{(j,i)}^\top y_{ji}$. The resulting AL dual function $L_a(\cdot)$, the (augmented) Lagrangian $L(\cdot)$, and the AL dual optimization problem are, respectively, given in (6.3), (6.4), and (6.5).

$$\begin{array}{rlll} L_a\left(\{\lambda_{(i,j)}\}, \{\eta_{(i,j)}\}\right) = & \min & L\left(\{x_i\}, \{y_{ij}\}, \{\lambda_{(i,j)}\}, \{\eta_{(i,j)}\}\right) & \\ & \text{subject to} & x_i \in \mathcal{X}_i,\ i = 1, ..., N & (6.3) \\ & & y_{ij} \in \mathbb{R}^d,\ (i,j) \in E_d & \end{array}$$

$$L\left(\{x_i\},\{y_{ij}\},\,\{\lambda_{(i,j)}\},\{\eta_{(i,j)}\}\right) \;=\; \sum_{i=1}^{N} f_i(x_i) + \sum_{(i,j)\in E_d} \eta_{(i,j)}^{\top}\,(x_i - y_{ij}) \tag{6.4}$$

$$+ \sum_{\{i,j\}\in E,\, i<j} \lambda_{(i,j)}^{\top}\, y_{ij} - \lambda_{(j,i)}^{\top}\, y_{ji} + \frac{1}{2}\rho \sum_{(i,j)\in E_d} \|x_i - y_{ij}\|^2 + \frac{1}{2}\rho \sum_{\{i,j\}\in E,\, i<j} \|y_{ij} - y_{ji}\|^2$$

$$\begin{array}{ll} \text{maximize} & L_a\left(\{\lambda_{(i,j)}\},\,\{\eta_{(i,j)}\}\right) \\ \text{subject to} & \lambda_{(i,j)} = \lambda_{(j,i)},\;\{i,j\}\in E\;\cdot \\ & \eta_{(i,j)}\in\mathbb{R}^d,\;(i,j)\in E_d \end{array} \tag{6.5}$$

In (6.4), ρ is a positive parameter. See [15] for some background on AL methods. The terms $\lambda_{(i,j)}^{\top}\, y_{ij} - \lambda_{(j,i)}^{\top}\, y_{ji}$ in the sum $\sum_{\{i,j\}\in E} \lambda_{(i,j)}^{\top}\, y_{ij} - \lambda_{(j,i)}^{\top}\, y_{ji}$ are arranged such that $i<j$, for all $\{i,j\}\in E$. [5]

Denote by $d^\star$ the optimal value of the dual problem (6.5), the dual of (6.2). Under Assumption 6.1, the strong duality between (6.2) and (6.5) holds, and $d^\star = f^\star$; moreover, the set of optimal solutions $\mathcal{D}^\star = \left\{\{\lambda^\star_{(i,j)}\},\,\{\eta^\star_{(i,j)}\} :\, L_a\left(\{\lambda^\star_{(i,j)}\},\,\{\eta^\star_{(i,j)}\}\right) = f^\star\right\}$ is nonempty. Denote by $C := \mathcal{X}_1\times\mathcal{X}_2\times ...\times\mathcal{X}_N\times(\mathbb{R})^{m(2M)}$ the constraint set in (6.3), i.e., the constraints in (6.2) that are not dualized. Let x_0 be a point in $\mathrm{ri}(\mathcal{X}_i)$, $i=1,...,N$ (see Assumption 1.3.) Then, a point $(\{x_{i,0}\},\{y_{ij,0}\})\in C$, where $x_{i,0}=x_0$, $y_{ij,0}=x_0$, belongs to $\mathrm{ri}(C)$, and it clearly satisfies all equality constraints in the primal problem (6.2); hence, it is a Slater point, and the above claims on strong duality hold, [92]. We remark that strong duality holds for any choice of $\rho\geq 0$ (but we are interested only in the case $\rho>0$,) and, moreover, the set of dual solutions $\mathcal{D}^\star$ does not depend on the choice of ρ, provided that $\rho\geq 0$ (see, e.g., [33], p.359.)

6.3.2 Solving the dual: D–AL–G (dual augmented Lagrangian gossiping) algorithm

We now explain how to solve the dual problem (6.5). First, we note that (6.5) is equivalent to the unconstrained maximization of $L'_a\left(\{\lambda_{\{i,j\}}\},\{\eta_{(i,j)}\}\right) = \min_{x_i\in\mathcal{X}_i, y_{ij}\in\mathbb{R}^d} L'\left(\{x_i\},\{y_{ij}\},\,\{\lambda_{\{i,j\}}\},\{\eta_{(i,j)}\}\right)$, where the function $L'\left(\{x_i\},\{y_{ij}\},\,\{\lambda_{\{i,j\}}\},\{\eta_{(i,j)}\}\right)$ is defined by replacing both $\lambda_{(i,j)}$ and $\lambda_{(j,i)}$ in $L(\cdot)$ (eqn. (6.4)) with $\lambda_{\{i,j\}}$, for all $\{i,j\}\in E$. The standard method of multipliers for the unconstrained maximization of $L'_a(\cdot)$ is given by:

$$\begin{aligned} \lambda_{\{i,j\}}(t+1) &= \lambda_{\{i,j\}}(t) + \rho\;\mathrm{sign}(j-i)\;\left(y^\star_{ij}(t) - y^\star_{ji}(t)\right),\;\{i,j\}\in E \\ \eta_{(i,j)}(t+1) &= \eta_{(i,j)}(t) + \rho\;\left(x^\star_i(t) - y^\star_{ij}(t)\right),(i,j)\in E_d. \end{aligned} \tag{6.6}$$

[5] For each link $\{i,j\}\in E$, the virtual nodes y_{ij} and y_{ji} (i.e., nodes i and j,) have to agree beforehand (in the network training period) which one takes the $+$ sign and which one takes the $-$ sign in $\lambda_{(i,j)}^{\top}\, y_{ij} - \lambda_{(j,i)}^{\top}\, y_{ji}$. In (6.4), for sake of notation simplicity, the distribution of $+$ and $-$ signs at each link $\{i,j\}$ is realized by the order of node numbers, where a distinct number in $\{1,...,N\}$ is assigned to each node. However, what matters is only to assign $+$ to one node (say i) and $-$ to the other, for each $\{i,j\}\in E$.

$$\Big(\{x_i^\star(t)\},\, \{y_{ij}^\star(t)\} \Big) \in \begin{array}{ll} \text{arg min} & L'\left(\{x_i\},\, \{y_{ij}\},\, \{\lambda_{\{i,j\}}(t)\},\, \{\eta_{(i,j)}(t)\}\right) \\ \text{subject to} & x_i \in \mathcal{X}_i,\, i = 1, ..., N \\ & y_{ij} \in \mathbb{R}^d,\ (i,j) \in E_d. \end{array} \tag{6.7}$$

Assigning a copy of $\lambda_{\{i,j\}}$ to both nodes i (the corresponding copy is $\lambda_{(i,j)}$) and j (the corresponding copy is $\lambda_{(j,i)}$), (6.6) immediately yields an algorithm to solve (6.5), given by:

$$\begin{aligned} \lambda_{(i,j)}(t+1) &= \lambda_{(i,j)}(t) + \rho\ \text{sign}(j-i)\ \left(y_{ij}^\star(t) - y_{ji}^\star(t)\right),\ (i,j) \in E_d \\ \eta_{(i,j)}(t+1) &= \eta_{(i,j)}(t) + \rho\ \left(x_i^\star(t) - y_{ij}^\star(t)\right),\ (i,j) \in E_d, \end{aligned} \tag{6.8}$$

where

$$\Big(\{x_i^\star(t)\},\, \{y_{ij}^\star(t)\} \Big) \in \begin{array}{ll} \text{arg min} & L\left(\{x_i\},\, \{y_{ij}\},\, \{\lambda_{(i,j)}(t)\},\, \{\eta_{(i,j)}(t)\}\right) \\ \text{subject to} & x_i \in \mathcal{X}_i,\, i = 1, ..., N \\ & y_{ij} \in \mathbb{R}^d,\ (i,j) \in E_d. \end{array} \tag{6.9}$$

(Note that $\Big(\{x_i^\star(t)\},\, \{y_{ij}^\star(t)\} \Big)$ is the same in (6.7) and (6.9).) According to (6.8), essentially, both nodes i and j maintain their own copy ($\lambda_{(i,j)}$ and $\lambda_{(j,i)}$, respectively) of the same variable, $\lambda_{\{i,j\}}$. It can be shown that, under Assumption 6.1, any limit point of the sequence $\Big(\{x_i^\star(t)\}, \{y_{ij}^\star(t)\}\Big)$, $t = 0, 1, ...$, is a solution of (6.2) (see, e.g., [37], Section 3.4); and the corresponding limit point of the sequence $x_i^\star(t)$, $t = 0, 1, ...$, is a solution of (6.1).

Before updating the dual variables as in (6.8), the nodes need to solve problem (6.9), with fixed dual variables, to get $\Big(\{x_i^\star(t)\}, \{y_{ij}^\star(t)\}\Big)$. We will explain in the next subsection, how the P–AL–G algorithm solves problem (6.9) in a distributed, iterative way, at a fast time scale $\{k\}$. We remark that P–AL–G terminates after a finite number of iterations k, and thus produces an inexact solution of (6.9). We will see that, after termination of the P–AL–G algorithm, an inexact solution for y_{ji} is available at node i; denote it by $y_{ji}^L(t)$. Denote, respectively, by $x_i^F(t)$ and $y_{ij}^F(t)$, the inexact solutions for x_i and y_{ij} at node i, after termination of P–AL–G. Then, the implementable update of the dual variables is:

$$\begin{aligned} \lambda_{(i,j)}(t+1) &= \lambda_{(i,j)}(t) + \rho\ \text{sign}(j-i)\ \left(y_{ij}^F(t) - y_{ji}^L(t)\right) \\ \eta_{(i,j)}(t+1) &= \eta_{(i,j)}(t) + \rho\ \left(x_i^F(t) - y_{ij}^F(t)\right). \end{aligned} \tag{6.10}$$

Note that the "inexact" algorithm in (6.10) differs from (6.8) in that it does not guarantee that $\lambda_{(i,j)}(t) = \lambda_{(j,i)}(t)$, due to a finite time termination of P–AL–G.

6.3.3 Solving the primal: P–AL–G algorithm

Given $\{\lambda_{(i,j)}(t)\}$, $\{\eta_{(i,j)}(t)\}$, we solve the primal problem (6.9) by a randomized, block-coordinate, iterative algorithm, that we refer to as P–AL–G. To simplify notation, we will write only $\lambda_{(i,j)}$ and $\eta_{(i,j)}$ instead of $\lambda_{(i,j)}(t)$, $\eta_{(i,j)}(t)$. We remark that $\lambda_{(i,j)}(t)$, $\eta_{(i,j)}(t)$ stay fixed while the optimization in (6.9) is done (with respect to x_i, y_{ij}.)

The block-coordinate iterative algorithm works as follows: at time slot k, the function in (6.4) is optimized with respect to a single block-coordinate, either x_i or y_{ij}, while other blocks are fixed. Such an algorithm for solving (6.9) admits distributed implementation, as we show next. Minimization of the function $L\left(\{x_i\},\{y_{ij}\},\,\{\lambda_{(i,j)}\},\{\eta_{(i,j)}\}\right)$ with respect to x_i, while the other coordinates x_j and y_{ij} are fixed, is equivalent to the following problem:

$$\begin{array}{ll}\text{minimize} & f_i(x_i) + \left(\overline{\eta}_i - \rho\,\overline{y}_i\right)^\top x_i + \frac{1}{2}\rho\, d_i \|x_i\|^2 \\ \text{subject to} & x_i \in \mathcal{X}_i \end{array}, \tag{6.11}$$

where $\overline{\eta}_i = \sum_{j\in\Omega_i} \eta_{(i,j)}$ and $\overline{y}_i = \sum_{j\in\Omega_i} y_{ij}$. Thus, in order to update x_i, the node i needs only information from its (virtual) neighbors. Minimization of the function $L\left(\{x_i\},\{y_{ij}\},\,\{\lambda_{(i,j)}\},\{\eta_{(i,j)}\}\right)$ with respect to y_{ij}, while the other coordinates x_j and y_{lm} are fixed, is equivalent to:

$$\begin{array}{ll}\text{minimize} & \eta_{(i,j)}^\top\,(x_i - y_{ij}) + \lambda_{(i,j)}^\top \text{sign}(j-i)\,(y_{ij} - y_{ji}) + \frac{1}{2}\rho\|x_i - y_{ij}\|^2 + \frac{1}{2}\rho\|y_{ij} - y_{ji}\|^2 \\ \text{subject to} & y_{ij} \in \mathbb{R}^d. \end{array} \tag{6.12}$$

Thus, in order to update y_{ij}, the corresponding virtual node needs only to communicate information with its neighbors in the network, y_{ji} and x_i. Physical communication is required only with y_{ji} (i.e., with physical node j.) The optimization problem (6.12) is an unconstrained problem with convex quadratic cost and admits the closed form solution:

$$y_{ij} = \frac{1}{2}y_{ji} + \frac{1}{2}x_i + \frac{1}{2\rho}\left(\eta_{(i,j)} - \text{sign}(j-i)\,\lambda_{(i,j)}\right). \tag{6.13}$$

Distributed implementation

We have seen that the block-coordinate updates in (6.11) and (6.13) only require neighborhood information at each node. We next give the distributed implementation of P–AL–G (see Algorithm 5) using the asynchronous communication model defined in Section 6.2.

Algorithm 5 Algorithm with gossiping for solving (6.9) (P–AL–G)

1: **repeat**
2: Wait for the tick of one of the clocks T_j^x, T_{ij}^y.
3: If clock T_{ij}^y ticks, node i transmits to node j the current value of y_{ij}.
If node j successfully receives y_{ij}, it updates the variable y_{ji} according to the equation (6.13).
4: If clock T_i^x ticks, node i updates the variable x_i by solving (6.11).
5: **until** a stopping criterion is met.

Simplified notation and an abstract model of the P–AL–G

We now simplify the notation and introduce an abstract model for the P–AL–G algorithm, for the purpose of convergence analysis in Section 6.4. Denote, in a unified way, by z_i, the primal variables x_i and y_{ij}, i.e., $z_i := x_i$, $i = 1, ..., N$, and $z_l := y_{ij}$, $l = N+1, ..., N+2M$, $(i,j) \in E_d$. Then, we can write the function in (6.4), viewed as a function of the primal variables, simply as $L(z)$, $L : \mathbb{R}^{m(N+2M)} \to \mathbb{R}$. Also, denote in a unified way the constraint sets $C_i := \mathcal{X}_i$, $i = 1, ..., N$, and $C_l := \mathbb{R}^d$, $l = N+1, ..., 2M+N$ (C_l, $l = N+1, ..., N+2M$; these sets correspond to the constraints on y_{ij}, $(i,j) \in E_d$.) Finally, define $C := C_1 \times C_2 \times ... \times C_{N+2M}$. Thus, the optimizations in (6.11) and (6.13) are simply minimizations of $L(z)$ with respect to a single (block) coordinate z_l, $l = 1, ..., 2M+N$. Recall the definition of $\zeta(k)$, $k = 1, 2, ...$ in section 6.2. Further, denote $P_i := \mathbb{P}(\zeta(k) = i) > 0$, $i = 0, 1, 2, ..., 2M+N$. Then, it is easy to see that the AL–G algorithm can be formulated as in Algorithm 6.

Finally, we summarize the overall dual AL–G algorithm in Algorithm 6.

Algorithm 6 AL–G algorithm at node i

1: Set $t = 0$, $\lambda_{(i,j)}(t=0) = 0$, $\eta_{(i,j)}(t=0) = 0$, $j \in \Omega_i$
2: **repeat**
3: Run P–AL–G (cooperatively with the rest of the network) to get $x_i^F(t)$, $y_{ij}^F(t)$ and $y_{ji}^L(t)$, $j \in \Omega_i$. At $k = 1, 2, ...$, if $\zeta(k) = i$, $i \neq 0$, then $z_i(k+1) = \arg\min_{w_i} L(\, z_1(k), ..., z_{i-1}(k), w, z_{i+1}(k), ..., z_{N+2M}(k)\,)$.
4: Update the dual variables, $\lambda_{(i,j)}(t)$, $\eta_{(i,j)}(t)$, $j \in \Omega_i$, according to eqn. (6.8).
5: Set $t \leftarrow t+1$
6: **until** a stopping criterion is met.

Remark. With AL–G, the updates of the primal variables, on a fast time scale k, are asynchronous and use

gossip communication, while the updates of the dual variables, on a slow time scale t, are *synchronous* and require no communication. Physically, this can be realized as follows. Each (physical) node in the network has a timer, and the timers of different nodes are synchronized. At the beginning of each (slow time scale) t-slot, nodes start the gossip communication phase and cooperatively run the P–AL–G algorithm. After a certain predetermined time elapsed, nodes stop the communication phase and, during an idle communication interval, they update the dual variables. After the idle time elapses, the nodes restart the communication phase at the beginning of the new t-slot.

Choice of ρ. It is known that, under Assumption 6.1, the method of multipliers (6.6) converges (i.e., any limit point of the sequence $x_i^\star(t)$, $t = 0, 1, ...$, is a solution of (6.1)) for any choice of the positive parameter ρ, [95], Theorem 2.1. It converges also if a sequence $\rho_{t+1} \geq \rho_t$ is used, [96], Proposition 4. See [15], 4.2.2, for a discussion on the choice of ρ. The method of multipliers still converges if we use different parameters $\rho = \rho_{(\lambda_{(i,j)},t)}$, $\rho = \rho_{(\eta_{(i,j)},t)}$, for each of the variables $\lambda_{(i,j)}$, $\eta_{(i,j)}$. This corresponds to replacing the quadratic terms $\rho \|x_i - y_{ij}\|^2$ and $\rho \|y_{ij} - y_{ji}\|^2$ in eqn. (6.4) with $\rho_{(\eta_{(i,j)},t)} \|x_i - y_{ij}\|^2$ and $\rho_{(\eta_{(i,j)},t)} \|y_{ij} - y_{ji}\|^2$, respectively. See reference [97] for details. (We still need $\rho_{(\lambda_{(i,j)},t)} \approx \rho_{(\lambda_{(j,i)},t)}$.) Equation (6.11) becomes[6]

$$\begin{array}{ll} \text{minimize} & f_i(x_i) + \left(\overline{\eta}_i - \sum_{j\in\Omega_i} \rho_{(\lambda_{(i,j)},t)}\, y_{ij}\right)^\top x_i + \frac{1}{2}\left(\sum_{j\in\Omega_i} \rho_{(\eta_{(i,j)},t)}\right) \|x_i\|^2 \\ \text{subject to} & x_i \in \mathcal{X}_i \end{array} \tag{6.14}$$

and equation (6.13) becomes

$$y_{ij} = \frac{\rho_{(\lambda_{(i,j)},t)}}{\rho_{(\lambda_{(i,j)},t)} + \rho_{(\eta_{(i,j)},t)}} (x_i + y_{ji}) + \frac{\eta_{(i,j)} - \text{sign}(j-i)\lambda_{(i,j)}}{\rho_{(\lambda_{(i,j)},t)} + \rho_{(\eta_{(i,j)},t)}}. \tag{6.15}$$

One possibility for adjusting the parameters $\rho_{(\eta_{(i,j)},t)}$ and $\rho_{(\lambda_{(i,j)},t)}$ in a distributed way is as follows. Each node i adjusts (updates) the parameters $\rho_{(\lambda_{(i,j),t})}$, $\rho_{(\eta_{(i,j),t})}$, $j \in \Omega_i$. We focus on the parameter $\rho_{(\lambda_{(i,j)},t)}$; other parameters are updated similarly. Suppose that the current time is t. Node i has stored in its memory the constraint violation at the previous time $t-1$ that equals $\epsilon_{(\lambda_{(i,j)},t-1)} = \|y_{ij}^F(t-1) - y_{ji}^L(t-1)\|$. Node i calculates the constraint violation at the current time $\epsilon_{(\lambda_{(i,j)},t)} = \|y_{ij}^F(t) - y_{ji}^L(t)\|$. If $\epsilon_{(\lambda_{(i,j)},t)}/\epsilon_{(\lambda_{(i,j)},t-1)} \leq \kappa_{(\lambda_{(i,j)})} < 1$, then the constraint violation is sufficiently decreased, and the parameter $\rho_{(\lambda_{(i,j)},t)}$ remains unchanged, i.e., node i sets $\rho_{(\lambda_{(i,j)},t)} = \rho_{(\lambda_{(i,j)},t-1)}$; otherwise, node i increases the parameter, i.e., it sets $\rho_{(\lambda_{(i,j)},t)} = \sigma_{(\lambda_{(i,j)})}\rho_{(\lambda_{(i,j)},t-1)}$. The constants $\kappa_{(\lambda_{(i,j)})} \in (0,1)$ and $\sigma_{(\lambda_{(i,j)})} > 1$ are local to node i.

[6]Reference [97] proves convergence of the method of multipliers with the positive definite matrix (possibly time-varying) penalty update, see eqn. (1.5) in [97]; the case of different (possibly time-varying) penalties assigned to different constraints is a special case of the matrix penalty, when the matrix is diagonal (possibly time-varying.)

6.4 Convergence analysis of the AL–G algorithm

This section analyzes convergence of the AL–G algorithm. Convergence of the multiplier method for the dual variable updates (on slow time scale $\{t\}$) in (6.8) is available in the literature, e.g., [15]. We remark that, in practice, P–AL–G runs for a finite time, producing an inexact solution of (6.9). This, however, does not violate the convergence of the overall dual AL–G scheme, as corroborated by simulations in Section 6.6. The P–AL–G algorithm for the primal variable update (on the fast time scale $\{k\}$) is novel, and its convergence requires a novel proof.

In Chapter 7, we shall analyze distributed AL methods in terms of the overall number of per-node communications (overall number of inner iterations) under a restricted class of costs and unconstrained problems. Hence, Chapter 7 analytically accounts for the finite time termination of the inner primal algorithms and the inexactness of the dual updates.

We proceed with the convergence analysis of P–AL–G. First, we state an additional assumption on the function $f(\cdot)$, and we state Theorem 6.4 on the almost sure convergence of P–AL–G.

Assumptions and statement of the result

Recall the equivalent definition of the P–AL–G and the simplified notation in 6.3.3. The P–AL–G algorithm solves the following optimization problem:

$$\begin{array}{ll} \text{minimize} & L(z) \\ \text{subject to} & z \in C \end{array} . \tag{6.16}$$

We will impose an additional assumption on the function $L(z)$, and, consequently, on the function $f(\cdot)$. First, we give the definition of a block-optimal point.

Definition 6.2 (Block-optimal point) A point $z^\bullet = \left(z_1^\bullet, z_2^\bullet, ..., z_{N+2M}^\bullet\right)$ is block-optimal for the problem (6.16) if: $z_i^\bullet \in \arg\min_{w_i \in C_i} L\left(z_1^\bullet, z_2^\bullet, ..., z_{i-1}^\bullet, w_i, z_{i+1}^\bullet, ..., z_{N+2M}^\bullet\right)$, $i = 1, ..., N + 2M$.

Assumption 6.3 If a point $z^\bullet$ is a block-optimal point of (6.16), then it is also a solution of (6.16).

Remark. Assumption 6.3 is valid if, e.g., $f_i(x) = k_i\|x\|_1 + W_i(x)$, $k_i \geq 0$, where $W_i : \mathbb{R}^d \to \mathbb{R}$ is a continuously differentiable, convex function, and $\|x\|_1 = \sum_{i=1}^N |x_i|$ is the l_1 norm of x, [14].

Define the set of optimal solutions $B = \{z^\star \in C : \ L(z^\star) = L^\star\}$, where $L^\star = \inf_{z\in C} L(z)$.[7] Further, denote by $\mathrm{dist}(b, A)$ the Euclidean distance of point $b \in \mathbb{R}^d$ to the set $A \subset \mathbb{R}^d$, i.e., $\mathrm{dist}(b, A) = \inf_{a\in A} \|a - b\|_2$, where $\|x\|_2$ is the Euclidean, l_2 norm. We now state the Theorem on almost sure (a.s.) convergence of the P–AL–G algorithm (Theorem 6.4,) after which we give some auxiliary Lemmas needed to prove Theorem 6.4.

Theorem 6.4 Let Assumptions 6.1 and 6.3 hold, and consider the optimization problem (6.16) (with fixed dual variables.) Consider the sequence $\{z(k)\}_{k=0}^{\infty}$ generated by the algorithm P–AL–G. Then:

1. $\lim_{k\to\infty} \mathrm{dist}\,(z(k), B) = 0, \ \text{a.s.}$

2. $\lim_{k\to\infty} L\,(z(k)) = L^\star, \ \text{a.s.}$

Auxiliary Lemmas

Let $i_C : \mathbb{R}^d \to \mathbb{R}\cup\{+\infty\}$ be the indicator function of the set C, i.e., $i_C(z) = 0$ if $z \in C$ and $+\infty$ otherwise. It will be useful to define the function $L + i_C : \ \mathbb{R}^{m(N+2M)} \to \mathbb{R} \cup \{+\infty\}$, $(L + i_C)(z) = L(z) + i_C(z)$. Thus, the optimization problem (6.16) is equivalent to the unconstrained minimization of $(L + i_C)(\cdot)$. The following Lemma establishes properties of the set of solutions B, the optimal value $L^\star$, and the function $(L + i_C)(\cdot)$.

Lemma 6.5 Let Assumption 6.1 hold. The functions $L(z)$ and $(L+i_C)(z)$ are coercive, $L^\star > -\infty$, and the set B is nonempty and compact.

Proof: The function $L(z)$ (given in eqn. (6.4)) is coercive. To see this, consider an arbitrary sequence $\{z(j)\}_{j=1}^{\infty}$, where $\|z(j)\| \to \infty$ as $j \to \infty$. We must show that $L(z(j)) \to \infty$. Consider two possible cases: 1) there is $i \in \{1, ..., N\}$ such that $\|x_i(j)\| \to \infty$; and 2) there is no $i \in \{1, ..., N\}$ such that $\|x_i(j)\| \to \infty$. For case 1), pick an i such that $\|x_i(j)\| \to \infty$; then $f_i(x_i(j)) \to \infty$, and hence, $L(z(j)) \to \infty$. In case 2), there exists a pair (i, l) such that $\|y_{il}\| \to \infty$; but then, as $x_i(j)$ is bounded, we have that $\|x_i(j) - y_{il}(j)\|^2 \to \infty$, and hence, $L(z(j)) \to \infty$. The function $(L + i_C)(z)$ is coercive because $(L + i_C)(z) \geq L(z)$, $\forall z$, and $L(z)$ is coercive. The function $(L + i_C)(z)$ is a closed[8] (convex) function, because $L(z)$ is clearly a closed function and $i_C(z)$ is a closed function because C is a closed set; hence, $(L + i_C)(z)$ is closed function as a sum of two closed functions.Hence, B is a closed set, as a sublevel set of the closed function $(L + i_C)(z)$.

[7] Under Assumption 6.1, the set B is nonempty and compact and $L^\star > -\infty$. This will be shown in Lemma 6.5. Clearly, $L^\star = L^\star\left(\{\lambda_{(i,j)}\}, \{\eta_{(i,j)}\}\right)$ and $B = B\left(\{\lambda_{(i,j)}\}, \{\eta_{(i,j)}\}\right)$ depend on the dual variables. For simplicity, we write only $L^\star$ and B.

[8] A function $q : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ is closed if its epigraph $\mathrm{epi}(q) = \{(x, v) : \ q(x) \leq v\}$ is a closed subset of $\mathbb{R}^{m+1}$.

The set B is bounded as a sublevel set of a coercive function $(L + i_C)(z)$. Hence, B is closed and bounded, and thus, compact. We have that $L^\star > -\infty$ (and B is non empty) as $L(z)$ is a continuous, convex, and coercive on $\mathbb{R}^{m(N+2M)}$. $\square$

Define $U_\epsilon(B) = \{z : \text{dist}(z, B) < \epsilon\}$, and let $V_\epsilon(B)$ be its complement, i.e., $V_\epsilon(B) = \mathbb{R}^d \backslash U_\epsilon(B)$. Further, denote by S and F the initial sublevel sets of L and $L+i_C$, respectively, i.e., $S = \{z : L(z) \leq L(z(0))\}$, and $F = \{z : \ (L + i_C)(z) \leq L(z(0))\} = S \cap C$, where $z(0) \in C$ is a feasible, deterministic, initial point. We remark that, given $z(0)$, any realization of the sequence $\{z(k)\}_{k=0}^{\infty}$ stays inside the set F. This is true because $L(z(k))$ is a nonincreasing sequence by the definition of the algorithm P–AL–G and because any point $z(k)$ is feasible. Define also the set $\Gamma(\epsilon) = F \cap V_\epsilon(B)$. We now remark that, by construction of the P–AL–G algorithm, the sequence of iterates $z(k)$ generated by P–AL–G is a Markov sequence. We are interested in the expected decrease of the function $L(\cdot)$ in one algorithm step, given that the current point is equal to $z(k) = z$:

$$\psi(z) = \mathbb{E}\left[L\left(z(k+1)\right) | z(k) = z\right] - L(z). \tag{6.17}$$

Denote by $L^i(z)$ the block-optimal value of the function $L(z)$ after minimization with respect to z_i:

$$L^i(z) = \min_{w_i \in C_i} L\left(z_1, z_2, ..., z_{i-1}, w_i, z_{i+1}, ..., z_{N+2M}\right) \tag{6.18}$$

We have, by the definition of P–AL–G, that (recall the definition of P_i above Algorithm 6:)

$$\psi\left(z\right) \ = \ \sum_{i=1}^{N+2M} P_i \left(L^i(z) - L(z)\right). \tag{6.19}$$

Define $\phi(z) = -\psi(z)$. From (6.19), it can be seen that $\phi(z) \geq 0$, for any $z \in C$. We will show that $\phi(z)$ is strictly positive on the set $\Gamma(\epsilon)$ for any positive ϵ.

Lemma 6.6

$$\inf_{z \in \Gamma(\epsilon)} \phi(z) = a(\epsilon) > 0 \tag{6.20}$$

We first show that $\Gamma(\epsilon)$ is compact and that L^i is continuous on $\Gamma(\epsilon)$ (latter proof is in Appendix D.)

Lemma 6.7 (Compactness of $\Gamma(\epsilon)$) The set $\Gamma(\epsilon)$ is compact, for all $\epsilon > 0$.

Proof: We must show that $\Gamma(\epsilon)$ is closed and bounded. It is closed because it is the intersection of the closed sets F and $V_\epsilon(B)$. It is bounded because $\Gamma(\epsilon) \subset F$, and F is bounded. The set F is bounded as a sublevel set of the coercive function $L + i_C$. The set F is closed as a sublevel set of the closed function

$L + i_C$. □

Lemma 6.8 (Continuity of L^i) The function $L^i : \Gamma(\epsilon) \to \mathbb{R}$ is continuous, $i = 1, ..., N + 2M$.

Proof: [Proof of Lemma 6.6] First, we show that $\phi(z) > 0$, for all $z \in \Gamma(\epsilon)$. Suppose not. Then, we have: $L^i(z) = L(z)$, for all i. This means that the point $z \in \Gamma(\epsilon)$ is block-optimal; Then, by Assumption 6.3, the point z is an optimal solution of (6.16). This is a contradiction and $\phi(z) > 0$, for all $z \in \Gamma(\epsilon)$. Consider the infimum in (6.20). The infimum is over the compact set and the function $\phi(\cdot)$ is continuous (as a scaled sum of continuous functions $L^i(\cdot)$); thus, by the Weierstrass theorem, the infimum is attained for some $z^\bullet \in \Gamma(\epsilon)$ and $\phi(z^\bullet) = a(\epsilon) > 0$. □

Proof of Theorem 6.4–1

Recall the expected decrease of the function $L(\cdot)$, $\psi(z)$. We have:

$$\mathbb{E}\left[\psi\left(z(k)\right)\right] = \mathbb{E}\left[\mathbb{E}\left[L\left(z(k+1)\right)|z(k)\right] - L\left(z(k)\right)\right] = \mathbb{E}\left[L\left(z(k+1)\right)\right] - \mathbb{E}\left[L\left(z(k)\right)\right]. \tag{6.21}$$

On the other hand, we have that $\mathbb{E}\left[\psi(z(k))\right]$ equals:

$$\mathbb{E}\left[\psi\left(z(k)\right)|z(k) \in \Gamma(\epsilon)\right]\mathbb{P}\left(z(k) \in \Gamma(\epsilon)\right) + \mathbb{E}\left[\psi\left(z(k)\right)|z(k) \notin \Gamma(\epsilon)\right]\mathbb{P}\left(z(k) \notin \Gamma(\epsilon)\right). \tag{6.22}$$

Denote by $p_k = \mathbb{P}\left(z(k) \in \Gamma(\epsilon)\right)$. Since $\psi\left(z(k)\right) \leq -a(\epsilon)$, for $z(k) \in \Gamma(\epsilon)$, and $\psi\left(z(k)\right) \leq 0$, for any $z(k)$, we have that: $\mathbb{E}\left[\psi\left(z(k)\right)\right] = \mathbb{E}\left[L(z(k+1))\right] - \mathbb{E}\left[L(z(k))\right] \leq -a(\epsilon)\, p_k$; summing up latter inequality for $j = 0$ up to $j = k - 1$, we get:

$$\mathbb{E}\left[L(z(k))\right] - L(z(0)) \leq -a(\epsilon)\sum_{j=0}^{k-1} p_k,\ \forall j \geq 0. \tag{6.23}$$

The last inequality implies that: $\sum_{k=0}^{\infty} p_k \leq \frac{1}{a(\epsilon)}\left(L\left(z(0)\right) - L^\star\right) < \infty$. Thus, by the first Borel-Cantelli Lemma, $\mathbb{P}\left(z(k) \in \Gamma(\epsilon),\ \text{infinitely often}\right) = 0$, $\forall \epsilon > 0$. Thus, $\mathbb{P}\left(\mathcal{A}_\epsilon\right) = 1$, $\forall \epsilon > 0$, where the event $\mathcal{A}_\epsilon$ is: $\mathcal{A}_\epsilon := \{\text{the tail of the sequence } z(k) \text{ belongs to } U_\epsilon(B)\}$. Consider the event $\mathcal{A} := \cap_{s=1}^{\infty}\mathcal{A}_{\epsilon_s}$, where ϵ_s is a decreasing sequence, converging to 0. Then, $\mathbb{P}\left(\mathcal{A}\right) = \mathbb{P}\left(\cap_{s=1}^{\infty}\mathcal{A}_{\epsilon_s}\right) = \lim_{s\to\infty}\mathbb{P}\left(\mathcal{A}_{\epsilon_s}\right) = \lim_{s\to\infty} 1 = 1$. Now, the event $\mathcal{B} := \{\lim_{k\to\infty}\text{dist}(z(k), B) = 0\}$ is equal to $\mathcal{A}$, and thus $\mathbb{P}\left(\mathcal{B}\right) = 1$.

Expected number of iterations for convergence: Proof of Theorem 6.4–2

Consider now the sets $\mathcal{U}_\epsilon(B) = \{z : L(z) \leq \epsilon + L^\star\}$ and $\mathcal{V}_\epsilon(B) = \mathbb{R}^d \backslash \mathcal{U}_\epsilon(B)$ and define the sets $\mathcal{F}$ and $\mathcal{G}(\epsilon)$ as $\mathcal{F} = C \cap S$ and $\mathcal{G}(\epsilon) = \mathcal{F} \cap \mathcal{V}_\epsilon(B)$. Similarly as in Lemmas 6.8, we can obtain that

$$\inf_{z \in \mathcal{G}(\epsilon)} \phi(z) = b(\epsilon) > 0. \tag{6.24}$$

We remark that, once $z(k)$ enters the set $\mathcal{U}_\epsilon(B)$ at $k = K_\epsilon$, it never leaves this set, i.e., $z(k) \in \mathcal{U}_\epsilon(B)$, for all $k \geq K_\epsilon$. Of course, the integer K_ϵ is random. In the next Theorem, we provide an upper bound on the expected value of K_ϵ (the time slot when $z(k)$ enters the set $\mathcal{U}_\epsilon(B)$,) thus giving a stopping criterion (in certain sense) of the algorithm P–AL–G.

Theorem 6.9 (Expected number of iterations for convergence) Consider the sequence $\{z(k)\}_{k=0}^{\infty}$ generated by the algorithm P–AL–G. Then, we have:

$$\mathbb{E}\left[K_\epsilon\right] \leq \frac{L\left(z(0)\right) - L^\star}{b(\epsilon)}. \tag{6.25}$$

Proof: Let us define an auxiliary sequence $\widetilde{z}(k)$ as $\widetilde{z}(k) = z(k)$, if $z(k) \in \mathcal{G}(\epsilon)$, and $z(k) = z^\star$, if $z(k) \in \mathcal{U}_\epsilon(B)$. Here $z^\star$ is a point in B. That is, $\widetilde{z}(k)$ is identical to $z(k)$ all the time while $z(k)$ is outside the set $\mathcal{U}_\epsilon(B)$ and $\widetilde{z}(k)$ becomes $z^\star$ and remains equal to $z^\star$ once $z(k)$ enters $\mathcal{U}_\epsilon(B)$. (Remark that $z(k)$ never leaves the set $\mathcal{U}_\epsilon(B)$ once it enters it by construction of Algorithm P–AL–G.)

Now, we have that:

$$\psi\left(\widetilde{z}(k)\right) = \begin{cases} \psi(z(k)) \leq -b(\epsilon) & \text{if } z(k) \in \mathcal{G}(\epsilon) \\ 0 & \text{if } z(k) \in \mathcal{U}_\epsilon(B) \end{cases}. \tag{6.26}$$

Taking the expectation of $\psi\left(z(k)\right)$, $k = 0, ..., t-1$ and summing up these expectations, and letting $t \to \infty$, we get:

$$\mathbb{E}\left[L\left(\widetilde{z}(\infty)\right)\right] - L(z(0)) = \sum_{k=0}^{\infty} \mathbb{E}\left[\psi\left(\widetilde{z}(k+1)\right)\right] - \mathbb{E}\left[\psi\left(\widetilde{z}(k)\right)\right] = \mathbb{E}\sum_{k=0}^{\infty} \psi\left(\widetilde{z}(k)\right) \leq -\mathbb{E}\left[K_\epsilon\right] b(\epsilon)$$

Thus, the claim in (6.25) follows. □

We now prove Theorem 6.4–2. By Theorem 10, the expected value of K_ϵ is finite, and thus K_ϵ is finite a.s. This means that for all $\epsilon > 0$, there exists random number K_ϵ (a.s. finite), such that $\widetilde{z}(k) = z^\star$, for all $k \geq K_\epsilon$, i.e., such that $z(k) \in \mathcal{U}_\epsilon(B)$ for all $k \geq K_\epsilon$. The last statement is equivalent to Theorem 6.4–2. □

6.5 Variants to AL–G: AL–MG (augmented Lagrangian multi neighbor gossiping) and AL–BG (augmented Lagrangian broadcast gossiping) algorithms

This section introduces two variants to the AL–G algorithm, the AL–MG (augmented Lagrangian multi neighbor gossiping) and the AL–BG (augmented Lagrangian broadcast gossiping). Relying on the previous description and analysis of the AL–G algorithm, the Section explains specificities of the AL–MG and AL–BG algorithms. Subsection 6.5.1 details the AL–MG, and subsection 6.5.2 details the AL–BG algorithm. Proofs of the convergence for P–AL–MG and P–AL–BG are in Appendix D.

6.5.1 AL–MG algorithm

The AL–MG algorithm is a variation of the AL–G algorithm. The algorithms AL–G and AL–MG are based on the same reformulation of (6.1) (eqn.(6.2)), and they have the same dual variable update (that is, D–AL–G and D–AL–MG are the same.) We proceed by detailing the difference between P–AL–MG and P–AL–G to solve (6.9) (with fixed dual variables.) With the algorithm P–AL–MG, each node has two independent Poisson clocks, T_i^x and T_i^y. Update followed by a tick of T_i^x is the same as with P–AL–G (see Algorithm 5, step 4.) If T_i^y ticks, then node i transmits *simultaneously* the variables y_{ij}, $j \in \Omega_i$, to all its neighbors (y_{i,j_1} is transmitted to node j_1, y_{i,j_2} is transmitted to node j_2, etc.) Due to link failures, the neighborhood nodes may or may not receive the transmitted information. Successfull transmissions are followed by updates of y_{ji}'s, according to (6.13). Define also the virtual clock T that ticks whenever one of the clocks T_i^x, T_i^y, ticks. Accordingly, we define the k-time slots as $[\tau_{k-1}, \tau_k)$, $k = 1, 2...$, $\tau_0 = 0$, and τ_k is the time of the k-th tick of T. Overall AL–MG algorithm is the same as AL–G (see Algorithm 6,) except that, instead of P–AL–G, nodes run P–AL–MG algorithms at each t. We prove convergence of the P–AL–MG in Appendix D; for convergence of the overall AL–MG algorithm, see discussion at the beginning of Section 6.5.

6.5.2 AL–BG algorithm: An algorithm for static networks

We now present a simplified algorithm for the networks with reliable transmissions. This algorithm is based on the reformulation of (6.1) that eliminates the variables y_{ij}'s. That is, we start with the following

equivalent formulation of (6.1):

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^N f_i(x_i) \\ \text{subject to} & x_i \in \mathcal{X}_i,\ i=1,...,N, \\ & x_i = x_j,\ \{i,j\} \in E \end{array} \tag{6.27}$$

We remark that (6.27) is equivalent to (6.1) because the supergraph is connected. After dualizing the constraints $x_i = x_j$, $(i,j) \in E$, the AL dual function $L_a(\cdot)$ and the Lagrangian $L(\cdot)$ become:

$$L_a\left(\{\lambda_{\{i,j\}}\}\right) = \begin{array}{ll} \min & L\left(\{x_i\},\, \{\lambda_{\{i,j\}}\}\right) \\ \text{subject to} & x_i \in \mathcal{X}_i,\ i=1,...,N \end{array}$$

$$L\left(\{x_i\},\, \{\lambda_{\{i,j\}}\}\right) = \sum_{i=1}^N f_i(x_i) + \sum_{\{i,j\}\in E,\, i<j} \lambda_{\{i,j\}}^\top\, (x_i - x_j) + \frac{1}{2}\rho \sum_{\{i,j\}\in E,\, i<j} \|x_i - x_j\|^2. \tag{6.28}$$

In the sums $\sum_{\{i,j\}\in E} \lambda_{\{i,j\}}^\top\, (x_i - x_j)$ and $\sum_{\{i,j\}\in E} \|x_i - x_j\|^2$, the terms $\lambda_{\{i,j\}}^\top\, (x_i - x_j)$ and $\|x_i - x_j\|^2$ are included once. (The summation is over the undirected edges $\{i,j\}$.) Also, terms $\lambda_{\{i,j\}}^\top\, (x_i - x_j)$ in the sum $\sum_{\{i,j\}\in E} \lambda_{\{i,j\}}^\top\, (x_i - x_j)$ are arranged such that $i < j$, for all $\{i,j\} \in E$. The resulting dual optimization problem is the unconstrained maximization of $L_a(\lambda_{\{i,j\}})$.

Solving the dual: D–AL–BG algorithm

We solve the dual (6.28) by the method of multipliers, which can be shown to have the following form:

$$\lambda_{\{i,j\}}(t+1) = \lambda_{\{i,j\}}(t) + \rho\, \mathrm{sign}(j-i)\, \left(x_i^\star(t) - x_j^\star(t)\right) \tag{6.29}$$

$$x^\star(t) = (x_1^\star(t), x_2^*(t), ..., x_N^*(t)) \in \begin{array}{ll} \text{arg min} & L\left(\{x_i\},\, \{\lambda_{\{i,j\}}(t)\}\right) \\ \text{subject to} & x_i \in \mathcal{X}_i,\ i=1,...,N \end{array}. \tag{6.30}$$

We will explain in the next paragraph how the P–AL–BG algorithm solves (6.30) in a distributed, iterative way. With AL–BG, each node needs to maintain only one m-dimensional dual variable: $\overline{\lambda}_i := \sum_{j\in\Omega_i} \mathrm{sign}(j-i)\, \lambda_{\{i,j\}}$. Also, define $\overline{x}_i := \sum_{j\in\Omega_i} x_j$. The P–AL–G algorithm terminates after a finite number of inner iterations k, producing an inexact solution. Denote by x_i^F (resp. x_j^F) the inexact solution of x_i (resp. x_j, $j \in \Omega_i$), available at node i, after termination of P–AL–BG. We will see that $x_i^F = x_i^L$,

$\forall i$; accordingly, after termination of P–AL–BG, node i has available $\overline{x}_i^F := \sum_{j\in\Omega_i} x_j^F$. Summing up equations (6.29) for $\lambda_{\{i,j\}}$, $j \in \Omega_i$, and taking into account the finite time termination of the P–AL–BG, we arrive at the following dual variable update at node i:

$$\overline{\lambda}_i(t+1) = \overline{\lambda}_i(t) + \rho \, \left(d_i \, x_i^F(t) - \overline{x}_i^F(t) \right), \; i = 1, ..., N. \tag{6.31}$$

Solving for (6.30)**: P–AL–BG algorithm**

We solve the problem (6.30) by a randomized, block-coordinate P–AL–BG algorithm. After straightforward calculations, it can be shown that minimization of the function in (6.28) with respect to x_i (while other coordinates are fixed) is equivalent to the following minimization:

$$\begin{array}{ll} \text{minimize} & f_i(x_i) + \left(\overline{\lambda}_i - \rho \, \overline{x}_i\right)^\top x_i + \frac{1}{2} \, \rho \, d_i \|x_i\|^2 \\ \text{subject to} & x_i \in \mathcal{X}_i \end{array} \tag{6.32}$$

Similarly as with AL–G, we assume that the clock ticks at all nodes are governed by independent Poisson process T_i's. P–AL–BG is as follows. Whenever clock T_i ticks, node i updates x_i via (6.32) and broadcasts the updated x_i to all the neighbors in the network. Discrete random iterations $\{k\}$ of the P–AL–BG algorithm are defined as ticks of the virtual clock T that ticks whenever one of T_i ticks. The P–AL–BG algorithm produces x_i^F and $\overline{x}_i^F$ at node i. Overall dual AL–BG algorithm is similar to the AL–G algorithm (see Algorithm 6), except that, at each t, nodes cooperatively run the P–AL–BG algorithm, instead of P–AL–G algorithm. We prove convergence of P–AL–BG in Appendix D; for convergence of the overall dual AL–BG scheme, see discussion at the beginning of Section 6.5.

6.6 Simulation examples

In this section, we consider two simulation examples, namely, l_1–regularized logistic regression for classification (subsection 6.6.1), and cooperative spectrum sensing for cognitive radio networks (subsection 6.6.2.) Both examples corroborate the convergence of our algorithms AL–G, AL–MG on random networks, and AL–BG on static networks, and illustrate tradeoffs that our algorithms show with respect to the existing literature. We compare our and existing algorithms with respect to: 1) communication cost; and 2) computational cost, while the communication cost is dominant in networked systems supported by wireless communication. On our simulation example, AL–BG converge faster than existing algorithms (in [23, 24, 98, 8])

[9]) on static networks in terms of communication cost, on both examples; at the same time, it has a larger computational cost. For our l_1-regularized logistic regression example and random networks, AL–G and AL–MG converges faster than existing algorithms ([23, 24][10]) in terms of communication cost, while having larger computational cost. For the cooperative spectrum sensing example and random networks, AL–G and AL–MG converge slower than existing algorithms [23, 24].

6.6.1 l_1–regularized logistic regression for classification

We study distributed learning of a linear discriminant function. In particular, we consider the l_1–regularized logistic regression optimization problem (eqn. (45) in [78]; see Subsections 7.1 and 10.2). We add private constraints and adapt the notation from [78] to fit our exposition.[11] The problem setup is as follows. Each node i, $i = 1, ..., N$, has N_d data samples, $\{a_{ij}, b_{ij}\}_{j=1}^{N_d}$, where $a_{ij} \in \mathbb{R}^d$ is a feature vector (data vector,) and $b_{ij} \in \{-1, +1\}$ is the class label of the feature vector a_{ij}. That is, when $b_{ij} = 1$ (respectively, -1,) then the feature vector a_{ij} belongs to the class "1" (respectively, "-1".) The goal is to learn the weight vector $w \in \mathbb{R}^d$, and the offset $v \in \mathbb{R}$, based on the available samples at all nodes, $\{a_{ij}, b_{ij}\}_{j=1}^{N_d}$, $i = 1, ..., N$, so that w is sparse, and the equality: $\text{sign}\left(a_{ij}^\top w + v\right) = b_{ij},\ i = 1, ..., N,\ j = 1, ..., N_d$, holds for the maximal possible number of data samples $\{a_{ij}, b_{ij}\}_{j=1}^{N_d}$, $i = 1, ..., N$. One approach to choose w and v is via l_1–regularized logistic regression; that is, choose $w^\star$ and $v^\star$ that solve the following optimization problem, [78]:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{N} \sum_{j=1}^{N_d} \log\left(1 + \exp\left(-b_{ij}(a_{ij}^\top w + v)\right)\right) + \lambda \|w\|_1 \\ \text{subject to} & w^\top w \leq k_i,\ i = 1, ..., N \\ & |v| \leq k_i',\ i = 1, ..., N \end{array} . \tag{6.33}$$

The parameter $\lambda > 0$ enforces the sparsity in w, [63]. The private constraints on w and v at node i (k_i's and k_i''s are positive) represent the prior knowledge available at node i (see [99], Chapter 7.) Problem (6.33) clearly fits our generic framework in (6.1) and has a vector optimization variable, a non smooth objective function, and quadratic private constraints. Alternatives to (6.33) to learn w and v include support vector machines and boosting, [63, 78].

[9] Reference [8] focusses specifically on the Lasso problem; we compare with [8] in subsection 6.6.2.

[10] Only references [23, 24] consider random networks.

[11] Note that [78] studies only the parallel network architecture, with a fusion center, and it does not propose an algorithm to solve the l_1–regularized logistic regression problem on generic networks, the case that we address here.

Simulation setup

We consider a supergraph with $N = 20$ nodes and $|E| = 37$ undirected edges (74 arcs). Nodes are uniformly distributed on a unit square and pairs of nodes with distance smaller than a radius r are connected by an edge. For networks with link failures, the link failures of different arcs at the same time slot are independent and the failures of the same arc at different time slots are independent also. Link failure probabilities π_{ij} are generated as follows: $\pi_{ij} = k\,\frac{\delta_{ij}^2}{r^2},\ \delta_{ij} < r$, where $k = 0.5$. Each node has $N_d = 5$ data samples. Each feature vector $a_{ij} \in \mathbb{R}^d$, $m = 20$, and the "true" vector w_{true} have approximately 60% zero entries. Nonzero entries of a_{ij} and w_{true}, and the offset v_{true} are generated independently, from the standard normal distribution. Class labels b_{ij} are generated by: $b_{ij} = \text{sign}\left(a_{ij}^\top w_{\text{true}} + v_{\text{true}} + \epsilon_{ij}\right)$, where ϵ_{ij} comes from the normal distribution with zero mean and variance 0.1. The penalty parameter λ is set to be $0.5 \cdot \lambda_{\max}$, where $\lambda_{\max}$ is the maximal value of λ above which the solution to (6.33) is $w^\star = 0$ (see ([78], subsection 10.2) how to find $\lambda_{\max}$.) We set k_i and k_i' as follows. We solve the unconstrained version of (6.33) via the centralized subgradient algorithm; we denote the corresponding solution by $w^\bullet$ and $v^\bullet$. We set $k_i = (1 + r_i) \cdot \|w^\bullet\|^2$, $k_i' = (1 + r_i') \cdot |v^\bullet|$, where r_i and r_i' are drawn from the uniform distribution on $[0, 1]$. Thus, the solution to problem (6.33) is in the interior of the constraint set. (Similar numerical results to the ones presented are obtained when the solution is at the boundary.) To update x_i with P–AL–G and P–AL–MG (6.11), we solve (6.11) via the projected subgradient algorithm.

Algorithms that we compare with

In the first set of experiments, we consider AL–BG for (static) networks; in the second set of experiments, we test AL–G and AL–MG on networks with link failures. We compare our algorithms with the ones proposed in [23, 2, 22, 24][12] and in [98]. References [23, 2, 22, 24] propose a primal projected subgradient algorithm, here refer to as PS (Primal Subgradient.) PS, as an intermediate step, computes weighted average of the optimal point estimates across node i's neighborhood. Averaging weights have not been recommended in [23, 2, 22, 24]; we use the standard time-varying Metropolis weights, see [65], (6.11). Reference [98] proposes an incremental primal subgradient algorithm, here referred to as MCS (Markov chain subgradient.) With MCS, the order of incremental updates is guided by a Markov chain, [98].[13] We simulate MCS and PS with fixed subgradient step size rather than the diminishing step size, as the former yields faster convergence.

We compare the algorithms based on two criteria. The first is the amount of inter-neighbor communica-

[12] We simulate the algorithms in [23, 2, 22, 24] with symmetric link failures.

[13] Convergence for MCS has been proved only with the projection onto a public constraint set, but we simulate it here with the straightforward generalization of the projection onto private constraint sets; MCS showed convergence for our example in the private constraints case also.

tion that the algorithms require to meet a certain accuracy. We count the total number of radio transmissions (counting both successful and unsuccessful transmissions.) The second is the total number of floating point operations (at all nodes.) In networked systems supported by wireless communication (e.g., WSNs,) the dominant cost (e.g., power consumption) is induced by communication. Total number of floating point operations depends on the algorithm implementation, but the results to be presented give a good estimate of the algorithms' computational cost. It may be possible to reduce the computational cost of AL–G, AL–MG, and AL–BG by a more computationally efficient solutions to problems (6.11) and (6.32) than (here adopted) projected subgradient method.

Denote by $f^\star$ the optimal value of (6.33). We compare the algorithms in terms of the following metric:

$$\mathbf{err}_f = \frac{1}{N}\sum_{i=1}^{N}\left(f(x_i) - f^\star\right),$$

where x_i is the estimate of the optimal solution available at node i at a certain time.

With our AL–G, AL-MG, and AL–BG algorithms, the simulations to be presented use an increasing sequence of AL penalty parameters (see the end of Section 6.4,) which, after some experimentation, we set to the following values: $\rho_t = t^{A_\rho} + B_\rho$, $t = 0, 1, ...$, with $A_\rho = 1.3$, and $B_\rho = 1$. We also implemented the algorithms with different and increasing ρ's assigned to each dual variable, with the scheme for adjusting ρ's explained at the end of Section 6.4, with $\kappa_{\lambda_{(i,j)}} = \kappa_{\eta_{(i,j)}} = 0.3$, and $\sigma_{\lambda_{(i,j)}} = \sigma_{\eta_{(i,j)}} = 1.2$. The latter choice also showed convergence of AL–G, AL-MG, and AL–BG, but the former yielded faster convergence. Our simulation experience shows that the convergence speed of AL–G, AL-MG, and AL–BG depend on the choice of ρ_t, but the optimal tuning of ρ_t is left for future studies. With our proposed methods, we set the number of inner iterations as follows. AL–G's t-th slow slot has $15900N$ inner iterations (ticks of T); AL–MG's $6000N + 1350tN$; and AL–BG's $500N$. With PS and MCS, and a fixed step size, the estimates $f(x_i)$ converge only to a neighborhood of $f^\star$. There is a tradeoff between the limiting error $\mathbf{err}_f(\infty)$ and the rate of convergence with respect to the stepsize α: larger α leads to faster convergence and larger $\mathbf{err}_f(\infty)$. We notice by simulation that AL–G, AL–MG, and AL–BG converge to a plateau neighborhood of $f^\star$; after that, they improve slowly; call the error that corresponds to this plateau $\mathbf{err}_f(ss)$. To make the comparison fair or in favor of PS and MCS, we set α for the PS and MCS algorithms such that the $\mathbf{err}_f(\infty)$ for PS and MCS is equal (or greater) than the $\mathbf{err}(ss)$ attained by AL–G, AL–MG, and AL–BG.

Results: Static network

Figure 6.2 (top left) plots $\mathbf{err}_f$ versus the number of ($m = 20$-dimensional vector) transmissions (cumulatively at all nodes.) We can see that AL–BG outperforms PS and MCS by one to two orders of magnitude. AL–BG needs about $0.3 \cdot 10^5$ transmissions to reduce $\mathbf{err}_f$ below 0.001, while MCS and PS need, respectively, about $4 \cdot 10^5$ and $18 \cdot 10^5$ transmissions for the same precision. With respect to the number of floating point operations (Figure 6.2, top right,) AL–BG needs more operations than MCS and PS; $45 \cdot 10^8$ for AL–BG versus $13 \cdot 10^8$ for PS, and $2 \cdot 10^8$ for MCS. Thus, with respect to MCS, AL–BG reduces communication at a cost of additional computation. Note that with AL–BG, MCS, and PS, due to private constraints, node i's estimate x_i may not be feasible at certain time slots; in this numerical example, AL–BG, MCS, and PS all produced feasible solutions at any time slot, at all nodes. A drawback of MCS in certain applications, with respect to PS and AL–BG, can be the *delay time* that MCS needs for the "token" to be passed from node to node as MCS evolves, see [98].

Results: Random network

Figure 6.2 (bottom left) plots $\mathbf{err}_f$ versus the total number of transmissions. AL–MG and AL–G converges faster than PS. To decrease $\mathbf{err}_f$ below $5 \cdot 10^{-4}$, AL–MG and AL–G require about $1.2 \cdot 10^6$ transmissions, and AL–G $1.5 \cdot 10^6$ transmissions; PS requires about $3.7 \cdot 10^6$ transmissions to achieve the same precision. Figure 6.2 (bottom right) plots $\mathbf{err}_f$ plots versus the total number of floating point operations. PS requires less computation than AL–G and AL–MG. To decrease $\mathbf{err}_f$ below $5 \cdot 10^{-4}$, AL–MG and AL–G require about $69 \cdot 10^9$ transmissions; PS requires about $2.8 \cdot 10^9$ transmissions for same precision. With each of the algorithms AL–G, AL–MG, and PS, each node i's estimate x_i was feasible along time slots.

6.6.2 Cooperative spectrum sensing for cognitive radio networks

We now consider cooperative spectrum sensing for cognitive radio networks. Cognitive radios are an emerging technology for improving the efficiency of usage of the radio spectrum. (For a tutorial on cognitive radios see, e.g., [100].) We focus here on the cooperative spectrum sensing approach that has been studied in [8, 5]. Suppose that N_r cognitive radios, located at x_r positions in 2D space, cooperate to determine: 1) the spatial locations; and 2) the power spectrum density (PSD) of primary users. Primary users can be located on N_s potential locations, x_s, on $\sqrt{N_s} \times \sqrt{N_s}$ square grid (See Figure 6.3, top, in [5].) For brevity, we omit the details of the problem setup; we refer to reference [8], subsection II-A, for the problem setup, and section II (eqn. (6.2)) in the same reference, for the Lasso optimization problem of estimating the locations and the

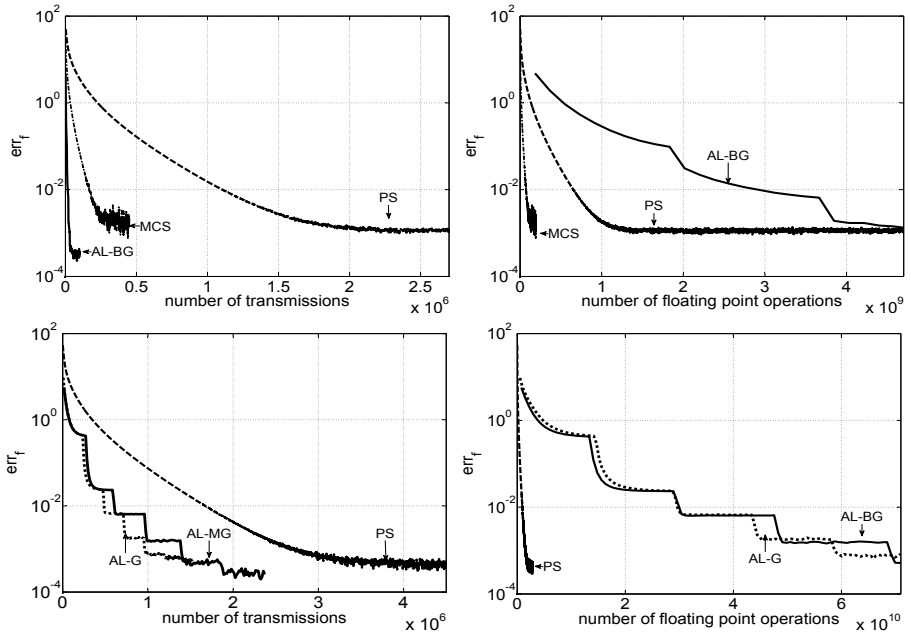

Figure 6.2: Performance of AL–BG, MCS, and PS on a static network (top figures,) and the AL–G, AL–MG and PS algorithms on a random network (bottom figures.) Left: total number of transmissions; Right: total number of floating point operations.

PSD of primary users. This (unconstrained) optimization problem in eqn. (6.2) in [8] fits the generic framework in eqn. (6.1); thus, our algorithms AL–G, AL–MG and AL–BG apply to solve the problem in (6.2) in [8]. Throughout, we use the same terminology and notation as in [8]. We now detail the simulation parameters. The number of potential sources is $N_s = 25$; they are distributed on a regular 5×5 grid over the square surface of 4km^2. Channel gains γ_{sr} are modeled as $\gamma_{sr} = \min\left\{1,\ \frac{A}{\|x_s - x_r\|^a}\right\}$, with $A = 200$ [meters] and a=3. The number of basis rectangles is $N_b = 6$, and the number of frequencies at which cognitive radios sample PSD is $N_f = 6$. There are 3 active sources; each source transmits at 2 out of $N_b = 6$ possible frequency bands. After some experimentation, we set the Lasso parameter λ (see (6.2) in [8]) to $\lambda = 1$; for a distributed algorithm to optimally set λ, see [8]. We consider the supergraph with $N_r = 20$ nodes (cognitive radios) and $|E| = 46$ undirected edges (92 arcs.) Nodes are uniformly distributed on a unit 2km$\times$2km square and the pairs of nodes with distance smaller than r =750m are connected.

For static networks, we compare AL–BG (our algorithm) with MCS, PS, and an algorithm in [8]. Reference [8] proposes three (variants of ADMM type algorithms, mutually differing in: 1) the total number of primal and dual variables maintained by each node (cognitive radio); 2) the method by which nodes solve local optimizations for primal variable update (These problems are similar to (6.32).) We compare AL–BG with the DCD-Lasso variant, because it has the same number of primal and dual variables as AL–BG and a smaller computational cost than the alternative DQP-Lasso variant. With AL–BG, we set the number of

inner iterations as follows: t-th slow time scale slot has $100N$ ticks of T (inner iterations). Further, with AL–BG, we use an increasing sequence of AL penalty parameters, $\rho_t = K_\rho A_\rho^t + C_\rho$, $t = 0, 1, ...$, with $K_\rho = 1$, $A_\rho = 1.15$ and $C_\rho = 3$. With DCD-Lasso, we use fixed $\rho = \rho_t$, as in [8, 5]. [14] After experimentation, we set $\rho = 8$. We solve the local problems in AL–BG (eqn. (6.32)), AL–G and AL–MG (eqn. (6.11),) by an efficient block coordinate method in [8] (see eqn. (13) in [8].) For the networks with link failures, we have compared our AL–G and AL–MG algorithms with PS (in [23, 2, 22, 24].) We briefly comment on the results. Both AL–G and AL–MG converge to a solution, in the presence of link failures as in 6.6.1; they converge slower than the PS algorithm, both in terms of communication and computational cost.

Results for static network

Figure 6.3 (left) plots $\mathbf{err}_f$ for PS, MCS, DCD-Lasso, and AL–BG versus the number of transmissions (at all nodes.) AL–BG shows improvement over the other algorithms. To achieve the precision of $\mathbf{err}_f \leq 0.044$, AL–BG requires about $5 \cdot 10^4$ transmissions; MCS $20 \cdot 10^4$ transmissions; DCD-Lasso $25 \cdot 10^4$ transmissions; PS $50 \cdot 10^4$ transmissions. Limiting error for PS is 0.027 (not visible in the plot.) Figure 6.3 (right) plots

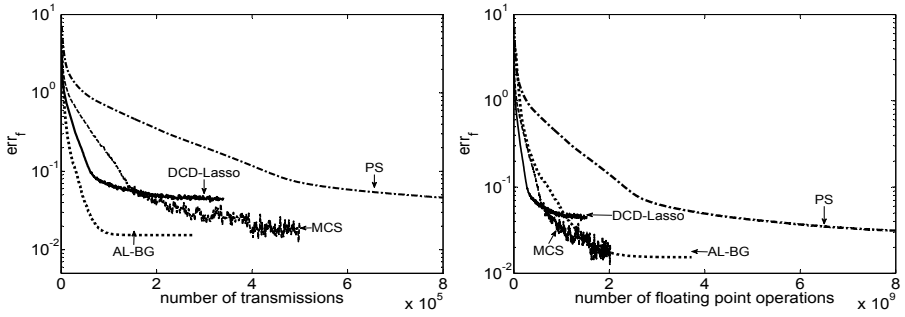

Figure 6.3: Performance of AL–BG, DCD-Lasso, PS and MCS algorithms on static CR network. Left: total number of transmissions (cumulatively, at all nodes). Right: total number of floating point operations (cumulatively, at all nodes.)

the $\mathbf{err}_f$ for the PS, MCS, DCD-Lasso, and AL–BG algorithms versus the total number of floating point operations. AL–BG, MCS and DCD-Lasso show similar performance, while PS is slower.

6.7 Conclusion

We considered very general distributed optimization problems with private, possibly nondifferentiable costs and private constraints. Nodes utilize gossip to communicate through a generic connected network with

[14] It may be possible to improve on the speed of DCD-Lasso by selecting appropriate time varying $\rho = \rho_t$; this is outside of our scope.

failing links. We proposed a novel distributed algorithm, the AL–G algorithm. AL–G handles private costs, private constraints, random networks, asymmetric link failures, and gossip communication.

This contrasts with existing augmented Lagrangian dual methods that handle only static networks and synchronous communication, while, as mentioned, the AL–G algorithm handles random networks and uses gossip communication. In distinction with existing distributed gradient algorithms that essentially handle symmetric link failures, AL–G handles asymmetric link failures.

AL–G updates the dual variables synchronously via a standard method of multipliers, and it updates the primal variables via a novel algorithm with gossip communication, the P–AL–G algorithm. P–AL–G is a nonlinear Gauss-Seidel type algorithm with random order of minimizations. Nonlinear Gauss-Seidel was previously shown to converge only under the *cyclic* or the *essentially cyclic* rules, [14, 15]; we prove convergence of P–AL–G, which has a *random* minimization order. Moreover, our proof is different from standard proofs for nonlinear Gauss-Seidel, as it uses as main argument the expected decrease in the objective function after one Gauss-Seidel step. We studied and proved convergence of two variants of AL–G, namely, AL–MG and AL–BG. An interesting future research direction is to develop a fully asynchronous dual algorithm that updates both the dual and primal variables asynchronously.

We demonstrated the effectiveness of our method on two simulation examples, l_1–regularized logistic regression for classification, and cooperative spectrum sensing for cognitive radios.

Chapter 7

Distributed Augmented Lagrangian Methods: Costs with Bounded Hessian

7.1 Introduction

In Chapter 6, we developed and analyzed distributed augmented Lagrangian (AL) dual methods for generic, nondifferentiable costs and private constraints. In this Chapter, we assume a restricted class of unconstrained problems with differentiable costs and bounded Hessian, but we establish strong convergence rate guarantees.

To state the problem, we recall the algorithm AL–BG from Chapter 6. Abstractly, this algorithm updates the primal variables $x_i(k)$ and dual variables $\eta_i(k)$ (at the outer iteration level) as follows:

$$(x_1(k+1), ..., x_N(k+1)) = \arg\min_{(x_1,...,x_N) \in \mathbb{R}^{dN}} L_a (x_1, ..., x_N; \eta_1(k), ..., \eta_N(k)) \tag{7.1}$$

$$\eta_i(k+1) = \eta_i(k) + \alpha \sum_{j \in O_i} W_{ij} \, (x_i(k+1) - x_j(k+1)) , \tag{7.2}$$

where $\alpha > 0$ is the (dual) step-size, and $L_a : \mathbb{R}^{dN} \times \mathbb{R}^{dN} \to \mathbb{R}$, is the AL function:

$$L_a(x_1, \, ... \, , x_N; \eta_1, ..., \eta_N) = \sum_{i=1}^{N} f_i(x_i) + \sum_{i=1}^{N} \eta_i^\top x_i + \frac{\rho}{2} \sum_{\{i,j\} \in E, \, i<j} W_{ij} \, \|x_i - x_j\|^2. \tag{7.3}$$

Differently from Chapter 6, we employ here a straightforward modification by weighting the (undirected) link $\{i, j\} \in E$ with the weight W_{ij}. Chapter 6 developed randomized, iterative methods, to solve (7.1) at a fast time scale. We shall consider here the primal methods from Chapter 6, as well as other (deterministic

and randomized) variants.

Specifically, we study a class of deterministic and randomized methods of type (7.1)–(7.2). Both deterministic and randomized methods update the dual variables via (7.2). With the deterministic variants, step (7.1) is done via multiple inner iterations of either: 1) the NJ method on $L(\cdot;\, \eta(k))$; or 2) the gradient descent on $L(\cdot;\, \eta(k))$. With both cases, one inner iteration corresponds to one per-node broadcast communication to all neighbors. With the randomized methods, step (7.1) is done either via multiple inner iterations of: 1) a randomized NGS method on $L(\cdot;\, \eta(k))$; or 2) a randomized coordinate gradient descent on $L(\cdot;\, \eta(k))$. Hence, we consider the total of four algorithm variants: 1) deterministic NJ; 2) deterministic gradient; 3) randomized NGS (this is AL–BG in [38]); and 4) randomized gradient. With all variants, we establish linear convergence rates in the total number of elapsed per-node communications $\mathcal{K}(k)$ after k outer iterations, assuming that nodes know beforehand (a lower bound on) $h_{\min}$, (an upper bound on) $h_{\max}$, and (a lower bound) on the network spectral gap λ_2.[1] (See [9] how this knowledge can be acquired in a distributed way.) With the deterministic variants, the distance to the solution $x^\star$ of (1.1), at any node i and any outer iteration k is upper bounded as:

$$\|x_i(k) - x^\star\| \leq \mathcal{R}^{\mathcal{K}(k)}\, \sqrt{N}\, \max\left\{ D_x,\, \frac{2\, D_\eta}{\sqrt{\lambda_2}\, h_{\min}} \right\}. \tag{7.4}$$

In (7.4), $\mathcal{R} \in [0, 1)$ is the convergence rate specified below, $D_x := \|x_i(0) - x^\star\|$ is the initial distance to the (primal) solution[2]; and 2) $D_\eta := \left(\frac{1}{N} \sum_{i=1}^N \|\nabla f_i(x^\star)\| \right)^2$.

With the randomized methods, we show the rate in the expected error norm:

$$\mathbb{E}\,[\, \|x_i(k) - x^\star\|\,] \leq \mathcal{R}^{\mathbb{E}[\,\mathcal{K}(k)\,]}\, \sqrt{N}\, \max\left\{ D_x,\, \frac{2\, D_\eta}{\sqrt{\lambda_2}\, h_{\min}} \right\}, \tag{7.5}$$

where $\mathbb{E}\,[\,\mathcal{K}(k)\,]$ is the expected number of elapsed per-node communications up to iteration k.

Table 7.1 shows the communication rates $\mathcal{R}$ for the four algorithm variants. (See ahead paragraph with heading Notation for the meaning of symbol Ω.) The quantity $\gamma := h_{\max}/h_{\min}$ is the condition number of the f_i's. We comment on the established rates. For example, for the deterministic NJ method, we can see that the rate $\mathcal{R}$ (the smaller the better) is jointly negatively affected by the condition number γ and the network "connectivity," measured by λ_2. For a poorly connected chain network of N nodes, $\lambda_2 = \Theta(1/N^2)$, and hence the rate is $1 - \Omega\left(\frac{1}{(1+\gamma)N^2}\right)$. In contrast, for well-connected expander networks, $\lambda_2 = \Omega(1)$, i.e., it stays bounded away from zero, and so the rate essentially does not deteriorate with the increase of

[1] The spectral gap λ_2 is the second smallest eigenvalue of the weighted Laplacian matrix $\mathcal{L} := I - W$.

[2] We assume throughout that all nodes start with the same initial point $x_i(0)$, say $x_i(0) = 0$.

	Convergence Rate $\mathcal{R}$
Deterministic, NJ:	$1-\Omega\left(\frac{\lambda_2}{1+\gamma}\right)$
Deterministic, Gradient:	$1-\Omega\left(\frac{\lambda_2}{1+\gamma}\,\frac{\log\left(1+\frac{1}{1+\gamma}\right)}{\log(1+\gamma)+\log\left(\lambda_2^{-1}\right)}\right)$
Randomized, NGS:	$1-\Omega\left(\frac{\lambda_2}{1+\gamma}\,\frac{1}{\log(1+\gamma)+\log\left(\lambda_2^{-1}\right)}\right)$
Randomized, Gradient:	$1-\Omega\left(\frac{\lambda_2}{(1+\gamma)^2}\,\frac{1}{\log(1+\gamma)+\log\left(\lambda_2^{-1}\right)}\right)$

Table 7.1: Convergence rates $\mathcal{R}$ in (7.4) (deterministic methods) and (7.5) (randomized methods) for the four variants of (7.1)–(7.2).

N. Further, comparing the communication rates $\mathcal{R}$ of the deterministic NJ and the deterministic gradient methods, we can see that the NJ variant has a slightly better dependence on the underlying network. This is natural, as the gradient method usually has a much smaller computational cost per-communication that the NJ method. Finally, we can see that deterministic NJ has a slightly better rate than the randomized NGS (a natural randomized counterpart), while the rates of the deterministic and randomized gradient methods are very similar, at least for a large γ.

It is worth noting that, with our deterministic and randomized gradient methods of type (7.1)–(7.2), both inner (primal) and outer (dual) iterations involve only calculations of the gradients ∇f_i's and weighted averaging of certain quantities across nodes' neighborhoods, just like the distributed methods in, e.g., [2, 12], and our Nesterov-based methods in Chapters 2 and 3. Although the methods in [9] achieve close-to-optimal convergence rates when the costs f_i's are *not strongly convex*, they do not converge linearly in the presence of strong convexity. In contrast, we show that the AL methods with gradient updates here achieve linear convergence, both in terms of the number of per-node communications and per-node gradient evaluations.

Our final comment is on the quantity $D_\eta := \left(\sum_{i=1}^N \|\nabla f_i(x^\star)\|\right)^{1/2}$ in (7.4) and (7.5) that arises from our analysis. This quantity measures, in a sense, the difficulty of (1.1) when solved by distributed methods like (7.1)–(7.2). The larger it is, the more difficult the problem is. If, in an extreme, the f_i's all have the same minimizer, say $y^\star$, then $y^\star$ is also the minimizer of (1.1) (We have $y^\star = x^\star$.) Such problem instance is "easy," because nodes do not need communication with other nodes to obtain the global solution to (1.1). Note that the "easyness" of the problem is in agreement with the value $D_\eta = 0$. On the other hand, if the nodes' local minimizers (of the f_i's), say $y_i^\star$'s, are very different, then $y_i^\star$ of a node i may be very different from $x^\star$. Hence, node i needs communication with other nodes to recover $x^\star$. This is in agreement with a large D_η in such scenarios. (See ahead Lemma 7.5 for the relation of D_η with the dual optimum.)

Brief comment on the literature. Augmented Lagrangian (AL) and alternating direction method of

multipliers (ADMM) methods have been studied for a long time; e.g., references [31, 32, 33] show locally linear or superlinear convergence rates of AL methods. Recently, there has been a strong renewed interest and progress in the convergence rate analysis of the classical AL and ADMM methods. References [34, 35] show that the ADMM method converges globally linearly, for certain more general convex costs than ours. These works are not concerned with distributed optimization over a generic network (1.1) that we consider here, but their results may imply linear convergence of the D-Lasso and related methods in, e.g., [8, 5]. Thus, the results in [34, 35] may imply linear convergence only for a *sub-class* of methods considered here. Further, our analysis is technically different from those in [34, 35]. Reference [36] analyzes the AL methods under more general costs than ours and is again not concerned with distributed optimization (1.1). The work [36] is related to ours in the sense that it analyzes the AL methods when the primal problems are solved inexactly, but, under their setup, the AL methods converge to a solution neighborhood. By contrast, in our setup, distributed AL methods converge linearly to the exact solution, in spite of the inexact solutions of the (inner) primal problems.

Reference [44] considers distributed optimization problem (1.1) over generic networks as we do here, under a wider class of functions than what we study. The reference shows $O\left(1/\mathcal{K}\right)$ rate of convergence in the number of per-node communications for a distributed ADMM method. Hence, with respect to our work, [44] studies a wider class of problems but establishes much slower rates. Reference [45] considers both the resource allocation problems and (1.1) and develops accelerated dual gradient methods. For problems (1.1), this reference gives the methods' local rates as $1-\Omega\left(\sqrt{\frac{\lambda_{\min}(AA^\top)}{\gamma\,\lambda_{\max}(AA^\top)}}\right)$, where A is the edge-node incidence matrix and $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ denote the minimal and maximal eigenvalues, respectively. Also, [45] considers ordinary dual problems, with the AL parameter $\rho = 0$; in contrast, we consider both the cases $\rho = 0$ and $\rho > 0$.

The works [28, 101, 102, 103] are related to our deterministic gradient AL variant. These references study distributed primal-dual methods that resemble our methods when the number of inner iterations τ is set to one (but are not the same.) These works do not establish convergence rates of their algorithms. Reference [104] studies a similar primal-dual method, based on the Arrow-Hurwitz-Uzawa method, in a centralized setting, and under more general costs than assumed here. The reference shows convergence to saddle point neighborhoods at rate $O(1/\sqrt{k})$ – a much slower rate than ours due to the assumed different (wider) function class.

Chapter organization. Section 7.2 details the network and optimization models that we assume, presents our deterministic AL distributed methods, and states our results on their convergence rates. Section 7.3 proves these results. Section 7.4 presents our randomized distributed AL methods and their rates,

while Section 7.5 proves these rates. Section 7.6 provides a simulation example with l_2-regularized logistic losses. Finally, we conclude in Section 7.7.

Notation. We denote by $\mathbb{R}^d$ the d-dimensional real coordinate space. Further, we denote by: a_l the l-th entry of vector a; A_{lm} or $[A]_{lm}$ the entry in the l-th row and m-th column of a matrix A; $A^\top$ the transpose of a matrix A; I, 0, 1, and e_i, respectively, the identity matrix, the zero matrix, the column vector with unit entries, and the i-th column of I; J the $N \times N$ ideal consensus matrix $J := (1/N)1\,1^\top$; $\|\cdot\|_l$ the vector (respectively, matrix) l-norm of its vector (respectively, matrix) argument; $\|\cdot\| = \|\cdot\|_2$ the Euclidean (respectively, spectral) norm of its vector (respectively, matrix) argument; $\lambda_i(\cdot)$ the i-th smallest eigenvalue; $A \succ 0$ means that the Hermitian matrix A is positive definite; $\lfloor a \rfloor$ the integer part of a real scalar a; $\nabla\phi(x)$ and $\nabla^2\phi(x)$ the gradient and Hessian at x of a twice differentiable function $\phi : \mathbb{R}^d \to \mathbb{R}$, $d \geq 1$; $\mathbb{P}(\cdot)$ and $\mathbb{E}[\cdot]$ the probability and expectation, respectively; and $\mathcal{I}(\mathcal{A})$ the indicator of event $\mathcal{A}$. For two positive sequences η_n and χ_n, $\eta_n = O(\chi_n)$ means that $\limsup_{n\to\infty} \frac{\eta_n}{\chi_n} < \infty$; $\eta_n = \Omega(\chi_n)$ means that $\liminf_{n\to\infty} \frac{\eta_n}{\chi_n} > 0$; and $\eta_n = \Theta(\chi_n)$ means that $\eta_n = O(\chi_n)$ and $\eta_n = \Omega(\chi_n)$.

The results in this Chapter are to be submitted in [48].

7.2 Deterministic Distributed Augmented Lagrangian Methods

Subsection 7.2.1 introduces the network and optimization models that we assume; Subsection 7.2.2 presents our two deterministic distributed AL methods, namely the method with the NGS primal updates, and the method with gradient-type primal updates. Subsection 7.2.3 states and interprets our convergence rates results.

7.2.1 Optimization and network models

Optimization model. We consider a distributed optimization problem where N nodes solve the unconstrained problem (1.1). The function $f_i : \mathbb{R}^d \to \mathbb{R}$ is known only by node i. We impose the following structure on the f_i's.

Assumption 7.1 (Optimization model) The functions $f_i : \mathbb{R}^d \mapsto \mathbb{R}$ are convex, twice continuously differentiable, and have bounded Hessian, i.e., there exist $0 < h_{\min} < h_{\max} < \infty$, such that, for all i:

$$h_{\min} I \preceq \nabla^2 f_i(x) \preceq h_{\max} I, \;\; \forall x \in \mathbb{R}^d. \tag{7.6}$$

Under Assumption 7.1, problem (1.1) is solvable and has the unique solution $x^\star$. Denote by $f^\star = \inf_{x\in\mathbb{R}^d} f(x) = f(x^\star)$ the optimal value. Further, Assumption 7.1 implies Lipschitz continuity of the ∇f_i's and strong convexity of the f_i's, i.e., for all i:

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq h_{\max}\|x-y\|,\ \ \forall x,y\in\mathbb{R}^d$$
$$f_i(y) \geq f_i(x) + \nabla f_i(x)^\top (y-x) + \frac{h_{\min}}{2}\|x-y\|^2,\ \ \forall x,y\in\mathbb{R}^d.$$

Communication model. We associate with problem (1.1) a network $\mathcal{V}$ of N nodes, described by the graph $\mathcal{G} = (\mathcal{V}, E)$, where $E \subset \mathcal{V}\times\mathcal{V}$ is the set of edges.

Assumption 7.2 (Network model) The graph $\mathcal{G}$ is connected, undirected, and simple (no self/multiple links.)

Weight matrix and weighted Laplacian. Assign to graph $\mathcal{G}$ a symmetric, stochastic (rows sum to one and all the entries are non-negative), $N\times N$ weight matrix W, with, for $i\neq j$, $W_{ij}>0$ if and only if, $\{i,j\}\in E$, and $W_{ii} = 1 - \sum_{j\neq i} W_{ij}$. Denote also $\widetilde{W} := W - J$. (See (7.3) for the role of W.) We require that W is positive definite and that $\lambda_{N-1}(W) < 1$. These requirements on the matrix W can be fulfilled by nodes without knowledge of any global network parameters; see Chapter 2. Also, denote by $\mathcal{L} := I - W$ the weighted graph Laplacian matrix. The quantity $\lambda_2(\mathcal{L}) = 1-\mu \in [0,1)$ (the larger it is, the better) is the network spectral gap and measures, in a sense, how well connected the network is. For example, for a chain N-node network, $\lambda_2(\mathcal{L}) = \Theta\left(\frac{1}{N^2}\right)$, while, for expander graphs, it stays bounded away from zero as N grows.

7.2.2 Deterministic Distributed Augmented Lagrangian Methods

We present two variants of deterministic distributed AL algorithms of type (7.1)–(7.2). Section 7.3 explains how we derive these methods. They mutually differ in step (7.1). Both methods solve (7.1) through the inner iterations, indexed by s, and perform (7.2) in the outer iterations, indexed by k. With the first variant, nodes update their primal variables via a nonlinear Jacobi (NJ) method on $L_a(\cdot\,;\eta(k))$ in (7.3); with the second variant, they use a gradient descent method on $L_a(\cdot\,;\eta(k))$. At outer iterations k, with both variants, nodes update the dual variables via the dual gradient ascent method (while the primal variables ar fixed).

The distributed AL algorithm with nonlinear Jacobi primal updates. We proceed with detailing the first algorithm variant. Later, to present the second variant, we only indicate the differences of the two methods. Denote by: $x_i(k,s)$ the node i's primal variable at the inner iteration s and outer iteration k; and $\eta_i(k)$ the node i's dual variable at the outer iteration k. Further, as in (7.1)–(7.2), denote by $x_i(k+1)$

the node i's primal variable at the end of k-th outer iteration. We make the following relation between the primal variables at the inner and outer iterations: $x_i(k, s = 0) := x_i(k)$, and $x_i(k + 1) := x_i(k, s = \tau)$.In addition, nodes maintain a weighted average of their own and the neighbors' primal variables $\overline{x}_i(k, s) := \sum_{j \in O_i} W_{ij}\, x_j(k, s)$. Recall that $O_i = \{j \in \{1, ..., N\} : W_{ij} > 0\}$ is the neighborhood set of node i, including node i.

The algorithm has, as the tuning parameters, the weight matrix W, the number of inner iterations per outer iteration τ, the AL penalty parameter $\rho \geq 0$, and the dual step-size $\alpha > 0$. The algorithm is given in Algorithm 7. Its operation is summarized as follows. At each inner iteration s, $s = 0, ..., \tau - 1$, each

Algorithm 7 Distributed deterministic AL with nonlinear Jacobi updates

1: (**Initialization**) Node i sets $k = 0$, $x_i(k = 0) \in \mathbb{R}^d$, $\overline{x}_i(k = 0) = x_i(0)$, and $\eta_i(k = 0) = 0$.
2: (**Inner iterations**) Node cooperatively run the nonlinear Jacobi method for $s = 0, 1, ..., \tau - 1$, with $x_i(k, s = 0) := x_i(k)$ and $\overline{x}_i(k, s = 0) := \overline{x}_i(k)$:

$$x_i(k, s+1) = \arg\min_{x_i \in \mathbb{R}^d} \left(f_i(x_i) + (\eta_i(k) - \rho\, \overline{x}_i(k, s))^\top x_i + \frac{\rho\, \|x_i\|^2}{2} \right) \tag{7.7}$$

$$\overline{x}_i(k, s+1) = \sum_{j \in O_i} W_{ij}\, x_j(k, s+1), \tag{7.8}$$

and set $x_i(k + 1) := x_i(k, s = \tau)$, $\overline{x}_i(k + 1) = \overline{x}_i(k, s = \tau)$.
3: (**Outer iteration**) Node i updates the dual variable $\eta_i(k)$ via:

$$\eta_i(k + 1) = \eta_i(k) + \alpha\, (x_i(k + 1) - \overline{x}_i(k + 1)). \tag{7.9}$$

4: Set $k \mapsto k + 1$ and go to step 2.

node i solves the local optimization problem (7.7), to obtain $x_i(k, s+1)$; then, node i broadcasts $x_i(k, s+1)$ to all its neighbors $j \in O_i - \{i\}$, as well as receives $x_j(k, s + 1)$, for all $j \in O_i - \{i\}$; upon reception of the $x_j(k, s + 1)$'s, node i computes $\overline{x}_i(k, s + 1)$ via (7.8). At outer iteration k, node i updates $\eta_i(k)$ via (7.9). (Note that (7.9) is equivalent to (7.2).) Note that each inner iteration requires one (d-dimensional) broadcast transmission per node, while the outer (dual) iterations do not require communication. Overall, node i performs τ broadcast transmissions per k.

To avoid notational clutter, we assume throughout, with all proposed algorithms, that all nodes use the same initial primal variable $x_i(0) = x_j(0)$, $\forall i, j$; for example, nodes can set $x_i(0) = 0$, $\forall i$.

The distributed AL algorithm with gradient-type primal updates. As noted, this algorithm variant is very similar to its alternative. The only difference is the following. In Algorithm 7, replace the nonlinear Jacobi update (7.7) with the gradient descent update on $L_a(\dot{;}\eta(k))$ in (7.3). After algebraic manipulations,

one obtains the following update at each node i:

$$x_i(k,s+1) = (1-\beta\,\rho)\;x_i(k,s) + \beta\,\rho\,\overline{x}_i(k,s) - \beta\;(\,\eta_i(k) + \nabla f_i(x_i(k,s))\,)\,, \tag{7.10}$$

where $\beta > 0$ is the (primal) step-size parameter. Hence, in addition to W, α, and ρ, current algorithm has an additional tuning parameter β.

7.2.3 Linear convergence: Statements of results

We are now ready to state our Theorem on the linear convergence of Algorithm 7 (deterministic NJ). [3] Recall the weighted Laplacian matrix $\mathcal{L} = I - W$, and its second largest eigenvalue $\lambda_2 = \lambda_2(\mathcal{L}) > 0$. Recall $D_x := \|x_1(0) - x^\star\|$, and $D_\eta := \left(\,\frac{1}{N}\sum_{i=1}^N \|\nabla f_i(x^\star)\|^2\,\right)^{1/2}$.

Theorem 7.3 (Convergence rate: Deterministic nonlinear Jacobi) Consider Algorithm 7 under Assumptions 7.1 and 7.2, and suppose that the algorithm and network parameters satisfy the following:

$$\alpha \;\leq\; h_{\min} \tag{7.11}$$

$$\left(\frac{\rho}{\rho+h_{\min}}\right)^{\tau} \;<\; \frac{1}{3}\,\frac{\lambda_2(\mathcal{L})\,h_{\min}}{\rho+h_{\max}}. \tag{7.12}$$

Then, at any node i, $x_i(k)$ generated by Algorithm 7 converges linearly (in the outer iterations k) to the solution $x^\star$, with rate:

$$r_{\mathrm{det,nj}} := \max\left\{\,\frac{1}{2} + \frac{3}{2}\left(\frac{\rho}{\rho+h_{\min}}\right)^{\tau},\;\left(1 - \frac{\alpha\,\lambda_2(\mathcal{L})}{\rho+h_{\max}}\right) + \frac{3\alpha}{h_{\min}}\left(\frac{\rho}{\rho+h_{\min}}\right)^{\tau}\right\} < 1, \tag{7.13}$$

and there holds:

$$\|x_i(k) - x^\star\| \leq (r_{\mathrm{det,nj}})^k\,\sqrt{N}\,\max\left\{D_x,\;\frac{2\,D_\eta}{\sqrt{\lambda_2(\mathcal{L})}\,h_{\min}}\right\}. \tag{7.14}$$

Condition 7.12 holds, for example, if $\rho \leq h_{\min}$, $\alpha = h_{\min}$, and:

$$\tau \geq \left\lceil \frac{\log\left(\frac{3(1+\gamma)}{\lambda_2(\mathcal{L})}\right)}{\log(2)} \right\rceil, \tag{7.15}$$

where $\gamma = h_{\max}/h_{\min}$ is the condition number of the f_i's. Thus, τ needs to grow only moderately with N. For a N-node chain network, $\lambda_2(\mathcal{L}) = \Theta(1/N^2)$, and hence it suffices to take $\tau = \Theta(\log N)$. The

[3] We shall state the Theorems on the rates for our four methods in terms of the outer iterations k, and subsequently we derive the communication rates in Table 7.1.

logarithmic growth of τ also suffices with any other type of a N-node connected network. For expander graphs, one can take $\tau = \Theta(1)$.

We make several remarks with respect to Theorem 7.3. First, the algorithm converges linearly in the *inner* iterations (number of per-node communications) as well, with the rate $\mathcal{R} := r^{1/\tau} = (r_{\mathrm{det,nj}})^{1/\tau}$. Taking:

$$\tau = \left\lceil \frac{\log\left(\frac{6(1+\gamma)}{\lambda_2(\mathcal{L})}\right)}{\log(2)} \right\rceil, \quad \alpha = \rho = h_{\min},$$

and using Taylor expansions, one obtains the communication rate $\mathcal{R}$ in Table 7.1. Second, for a fixed $h_{\min}$, $h_{\max}$, $\mathcal{L}$, and $\rho > 0$, there is a tradeoff with respect to the choice of τ. Increasing τ decreases (improves) the convergence rate r at the outer iteration level, but increases (deteriorates) the convergence rate at the inner iteration level (as $r^{1/\tau}$.) Finally, it is worth noting that, although $\rho = 0$ gives the best *upper bound* in (7.75), it may not correspond to the optimal *actual rate*. Indeed, reference [105] considers the special case of consensus problem, where each $f_i(x) : \mathbb{R} \to \mathbb{R}$, is of the form $f_i(x) = \frac{1}{2}(x - a_i)^2$, $a_i \in \mathbb{R}$, and a distributed ADMM method, which corresponds to a positive ρ and $\tau = 1$ in our setting. The reference shows that it is optimal to take the (nonzero) $\rho = \Theta\left(\frac{1}{\lambda_2(\mathcal{L})}\right)$, in which case the rate r is $1 - \Omega\left(\sqrt{\lambda_2(\mathcal{L})}\right)$; in contrast, for $\rho = 0$, the rate r is poorer and is $1 - \Omega\left(\lambda_2(\mathcal{L})\right)$. An interesting research direction is to address the optimal tuning of the parameters τ and ρ for the generic f_i's.

We now consider the variant with gradient primal updates and establish its linear convergence rate.

Theorem 7.4 (Convergence rate: Deterministic gradient updates) Consider Algorithm 7 where step (7.7) is replaced with (7.10), and let Assumptions 7.1 and 7.2 hold. Further, suppose that the algorithm and network parameters satisfy the following:

$$\alpha \leq h_{\min} \tag{7.16}$$

$$\beta \leq \frac{1}{h_{\max} + \rho} \tag{7.17}$$

$$(1 - \beta\, h_{\min})^{\tau} < \frac{1}{3}\, \frac{\lambda_2\, h_{\min}}{\rho + h_{\max}}. \tag{7.18}$$

Then, at any node i, $x_i(k)$ converges linearly (in the outer iterations k) to the solution $x^\star$, with rate:

$$r_{\mathrm{det,grad}} := \max\left\{ \frac{1}{2} + \frac{3}{2}\,(1 - \beta\, h_{\min})^{\tau},\ \left(1 - \frac{\alpha\, \lambda_2(\mathcal{L})}{\rho + h_{\max}}\right) + \frac{3\alpha}{h_{\min}}\,(1 - \beta\, h_{\min})^{\tau} \right\} < 1, \tag{7.19}$$

and there holds:

$$\|x_i(k) - x^\star\| \leq (r_{\mathrm{det,grad}})^k \sqrt{N} \max\left\{D_x,\ \frac{2\,D_\eta}{\sqrt{\lambda_2(\mathcal{L})}\,h_{\min}}\right\}. \tag{7.20}$$

Condition 7.18 holds, for example, if $\rho \leq h_{\min}$, $\beta = \frac{1}{h_{\max}+\rho}$, and: and:

$$\tau \geq \left\lceil \frac{\log\left(\frac{3(1+\gamma)}{\lambda_2(\mathcal{L})}\right)}{\log\left(\frac{\gamma+1}{\gamma}\right)} \right\rceil. \tag{7.21}$$

We comment on Theorem 7.4. First, the Theorem assures a linear convergence in the inner iterations (number of per-node communications) as well, with rate $\mathcal{R}_{\mathrm{det,grad}} = (r_{\mathrm{det,grad}})^{1/\tau}$. Taking:

$$\tau = \left\lceil \frac{\log\left(\frac{6(\gamma+1)}{\lambda_2(\mathcal{L})}\right)}{\log\left(\frac{\gamma+1}{\gamma}\right)} \right\rceil, \quad \alpha = \rho = h_{\min}, \quad \beta = \frac{1}{\rho + h_{\max}},$$

and using Taylor expansions, one obtains the communication rate in Table 7.1. The method has computationally simple (inner and outer) iterations, involving only gradient calculations and certain weighted averaging across nodes' neighborhoods. To our best knowledge, this is the first linear convergence rate result established for problems of type (1.1) (and the costs as general as given by Assumption 7.1), for the methods that involve only simple calculations (as opposed to the methods that involve local optimizations at each node, as, e.g., Algorithm 7.) Indeed, the gradient based methods in, e.g., [2], [9], as well as our methods in Chapters 2 and 3, require *diminishing* step-sizes for convergence to the exact solution. As a consequence, they do not achieve linear convergence rates on the functions class defined by Assumption 7.1. The AL method with gradient updates studied here converges to the exact solution under constant step-sizes α and β, which enables a linear convergence.

7.3 Convergence rate analysis: Proofs of Theorems 7.3 and 7.4

The goal of the current Section is to prove Theorems 7.3 and 7.4. The section is organized as follows. Subsection 7.3.1 sets up the analysis by introducing certain helpful objects and giving an alternative, compact representation of Algorithm 7 (see ahead Algorithm 8). Subsection 7.3.2 states and proves certain auxiliary Lemmas and finally proves Theorem 7.3.

7.3.1 Setting up analysis

For notational simplicity (to avoid extensive use of Kronecker products) we assume $d = 1$ (scalar optimization variable), but our analysis extends to a generic $d > 1$. We base our analysis on the following nonlinear saddle point system of equations:

$$\nabla F(x) + \eta + \rho\, \mathcal{L}\, x = 0 \tag{7.22}$$

$$\mathcal{L}x = 0 \tag{7.23}$$

$$1^\top \eta = 0. \tag{7.24}$$

In (7.22), $\rho \geq 0$ is the AL penalty parameter, and $F : \mathbb{R}^N \mapsto \mathbb{R}$ is defined by $F(x) = F(x_1, ..., x_N) = f_1(x_1) + f_2(x_2) + ... + f_N(x_N)$. In (7.22), $x \in \mathbb{R}^N$ is the primal variable, while $\eta \in \mathbb{R}^N$ is the dual variable; the i-th coordinate of x and η correspond to node i's primal and dual variables, respectively. The following Lemma demonstrates that solving (7.22) actually provides the solution to (1.1) at each node i.

Lemma 7.5 Consider (1.1), let Assumptions 7.1 and 7.2 hold, and consider the nonlinear system (7.22). Then, there exists a unique $(x^\bullet, \eta^\bullet) \in \mathbb{R}^N \times \mathbb{R}^N$ that satisfies (7.22)–(7.24), with $x^\bullet = x^\star 1$, where $x^\star$ is the solution to (1.1), and $\eta^\bullet = -\nabla F(x^\star\, 1)$.

We first show that $x^\bullet = x^\star 1$ and $\eta^\bullet = -\nabla F(x^\star 1)$ is a solution to (7.22)–(7.24). Consider (7.23). We have $\mathcal{L}x^\bullet = x^\star\, \mathcal{L}1 = 0$ (by the property of the Laplacian matrix on a connected network that zero is an eigenvalue with multiplicity one, and the corresponding eigenvector is 1.) Next, consider (7.24). We have:

$$1^\top \eta^\bullet = -\sum_{i=1}^{N} \nabla f_i(x^\star) = 0,$$

where the right equality holds because $x^\star$ is the solution to (1.1). Finally, consider (7.22). The equality holds because $\mathcal{L}x^\bullet = 0$ (already shown), and $\nabla F(x^\bullet) = \nabla F(x^\star 1) = -\eta^\bullet$. Thus, $(x^\bullet = x^\star 1,\ \eta^\bullet = -\nabla F(x^\star 1))$ satisfy (7.22)–(7.24).

Now, we show uniqueness. Suppose that (x', η') satisfy (7.22)–(7.24). We show that there has to hold: $x' = x^\star$, and $\eta' = -\nabla F(x^\star 1)$. By (7.23) and Assumption 7.2, x' is of the form $x' = \delta\, 1$, $\delta \in \mathbb{R}$. (This is a standard result on the Laplacian matrix of a connected network, e.g., [106].) Next, multiplying (7.22) from the left by $1^\top$, using (7.24), and $1^\top \mathcal{L} = 0$, the following has to hold:

$$1^\top \nabla F(x') = \sum_{i=1}^{N} \nabla f_i(\delta) = \nabla f(\delta) = 0.$$

Thus, $\delta = x^\star$, because (1.1) has the unique solution by Assumption 7.1. Thus, $x' = x^\star 1$. Finally, by (7.22), we have $\eta' = -\nabla F(x') = -\nabla F(x^\star 1)$. Thus, $(x^\bullet = x^\star 1, \eta^\bullet = -\nabla F(x^\star 1))$ is the unique point that satisfies (7.22)–(7.24). Next, introduce the following two maps $\Phi : \mathbb{R}^N \mapsto \mathbb{R}^N$, and $\Psi : \mathbb{R}^N \mapsto \mathbb{R}^N$:

$$\Phi(x) := \nabla F(x) + \rho\, I\, x \tag{7.25}$$

$$\Psi(x) := \nabla F(x) + \rho\, \mathcal{L}\, x. \tag{7.26}$$

Further, define the maps: $\Phi^{-1} : \mathbb{R}^N \to \mathbb{R}^N$ and $\Psi^{-1} : \mathbb{R}^N \to \mathbb{R}^N$ by:

$$\Phi^{-1}(\eta) := \arg\min_{y \in \mathbb{R}^N} \left(F(y) - \eta^\top y + \frac{\rho}{2}\|y\|^2 \right) \tag{7.27}$$

$$\Psi^{-1}(\eta) := \arg\min_{y \in \mathbb{R}^N} \left(F(y) - \eta^\top y + \frac{\rho}{2} y^\top \mathcal{L} y \right). \tag{7.28}$$

The cost function in (7.28) is precisely L_a in (7.3). For any $\eta \in \mathbb{R}^N$, the above maps are well-defined by Assumption 7.1 (The latter assumption ensures that there exists a unique solution in the minimizations in (7.27) and (7.28), as the costs in (7.27) and (7.28) are strongly convex.) Next, we have:

$$\nabla F(\Phi^{-1}(\eta)) + \rho\, I \Phi^{-1}(\eta) = \eta = \Phi(\Phi^{-1}(\eta)),$$

where the left equality is by the first order optimality conditions, from (7.27), and the right equality is by definition of Φ in (7.25). Thus, the map Φ^{-1} is the inverse of Φ. Likewise, the map Ψ^{-1} is the inverse of Ψ. By the inverse function theorem, e.g., [107], the maps $\Phi^{-1} : \mathbb{R}^N \to \mathbb{R}^N$ and $\Psi^{-1} : \mathbb{R}^N \to \mathbb{R}^N$ are continuously differentiable, with the derivatives:

$$\nabla \Phi^{-1}(\eta) = \left(\nabla^2 F(\Phi^{-1}(\eta)) + \rho\, I \right)^{-1} \tag{7.29}$$

$$\nabla \Psi^{-1}(\eta) = \left(\nabla^2 F(\Psi^{-1}(\eta)) + \rho\, \mathcal{L} \right)^{-1}. \tag{7.30}$$

Using the following identity for a continuously differentiable map $h : \mathbb{R}^N \to \mathbb{R}^N$:

$$h(u) - h(v) = \left[\int_{z=0}^{1} \nabla h(v + z(u-v)) dz \right] (u - v), \ \ \forall u, v \in \mathbb{R}^N, \tag{7.31}$$

we obtain the following useful relations:

$$\begin{aligned}
\Phi^{-1}(\eta_1) - \Phi^{-1}(\eta_2) &= R_\Phi(\eta_1, \eta_2)\,(\eta_1 - \eta_2) && (7.32)\\
R_\Phi(\eta_1, \eta_2) &:= \int_{z=0}^{1} \nabla\Phi^{-1}(\eta_1 + z(\eta_2 - \eta_1))\,dz \quad (\in \mathbb{R}^{N\times N})\\
\Psi^{-1}(\eta_1) - \Psi^{-1}(\eta_2) &= R_\Psi(\eta_1, \eta_2)(\eta_1 - \eta_2) && (7.33)\\
R_\Psi(\eta_1, \eta_2) &:= \int_{z=0}^{1} \nabla\Psi^{-1}(\eta_1 + z(\eta_2 - \eta_1))\,dz \quad (\in \mathbb{R}^{N\times N}).
\end{aligned}$$

By Assumption 7.1, we have that: $h_{\min}\, I \preceq \nabla^2 F(x) \preceq h_{\max}\, I$, $\forall x \in \mathbb{R}^N$. Using the latter, (7.29), (7.30), and (7.31), and $\mathcal{L} = I - W$, $0 \preceq \mathcal{L} \preceq I$ ($W \succ 0$ by assumption and W is symmetric, stochastic), we obtain the following properties of the $N \times N$ matrices $R_\Phi(\eta_1, \eta_2)$ and $R_\Psi(\eta_1, \eta_2)$:

$$\begin{aligned}
\frac{1}{h_{\max} + \rho}\, I \;&\preceq\; R_\Phi(\eta_1, \eta_2) \preceq \frac{1}{h_{\min} + \rho}\, I, \quad \forall \eta_1, \eta_2 \in \mathbb{R}^N && (7.34)\\
\frac{1}{h_{\max} + \rho}\, I \;&\preceq\; R_\Psi(\eta_1, \eta_2) \preceq \frac{1}{h_{\min}}\, I, \quad \forall \eta_1, \eta_2 \in \mathbb{R}^N. && (7.35)
\end{aligned}$$

We now introduce a compact representation of Algorithm 8. Denote by $x(k) := (x_1(k), ..., x_N(k))^\top \in \mathbb{R}^N$ the vector that stacks the primal variables at outer iteration k at all nodes. Similarly, let $x(k, s) := (x_1(k, s), ..., x_N(k, s))^\top$, and $\eta(k) := (\eta_1(k), ..., \eta_N(k))^\top$. Then, using the definition of maps Φ^{-1} and Ψ^{-1} in (7.25) and (7.26), we obtain in Algorithm 8 an equivalent representation. Note that the variables

Algorithm 8 Distributed AL algorithm with NJ updates: Compact representation

1: Initialization: Set $k = 0$, $x(k = 0) = 0$, and $\eta(k = 0) = 0$.

2: (Inner iterations) Set $x(k, s = 0) := x(k)$; for $s = 0, 1, ..., \tau - 1$, perform the following:

$$x(k, s+1) \;=\; \Phi^{-1}\left(\,\rho\, W\, x(k, s) - \eta(k)\,\right); \qquad (7.36)$$

and set $x(k + 1) := x(k, s = \tau)$.

3: (Outer iteration) Update the dual variable $\eta(k)$ via:

$$\eta(k+1) = \eta(k) + \alpha\, \mathcal{L}\, x(k+1). \qquad (7.37)$$

4: Set $k \mapsto k + 1$ and go to step 2.

$\overline{x}_i(k, s)$ and $\overline{x}_i(k)$ are redundant, and we can eliminate them from Algorithm 8. This is due to the fact that all nodes use the same initial $x_i(0) = x_1(0)$, $\forall i$, and so they know $x_j(k = 0, s = 0)$, $j \in O_i$, to implement step (7.36) at $k = 0, s = 0$.

7.3.2 Auxiliary Lemmas and proof of Theorem 7.3

We outline current subsection. We view Algorithm 8 as an inexact version of (7.1)–(7.2). Lemma 7.6 gives a bound on this inexactness. Further, Lemma 7.7 upper bounds the primal error, while Lemma 7.8 gives an upper bound on the dual error. We close the subsection by proving Theorems 7.3 and 7.4 using the established Lemmas.

Denote by $\widetilde{x}(k) := x(k) - x^\bullet$ and $\widetilde{\eta}(k) := \eta(k) - \eta^\bullet$ the primal and dual errors, respectively. (Recall the primal and dual solutions $x^\bullet$ and $\eta^\bullet$ in Lemma 7.5.)

For the purpose of convergence rate analysis, we introduce an auxiliary sequence $x'(k) \in \mathbb{R}^N$, $k = 1, 2, ...$, defined by:

$$x'(k+1) = \Psi^{-1}\left(\eta(k)\right), \tag{7.38}$$

where $\eta(k)$ is the dual variable generated by Algorithm 8. Note that $x'(k+1)$ is precisely the *exact* solution in (7.1). If, for a fixed $\eta(k)$, we performed an infinite number of inner iterations in (7.36), then $x(k,s)$ would converge to $x'(k+1)$ as $s \to \infty$. However, we terminate the inner algorithm after a finite number of τ inner iterations, and hence $x(k+1) = x(k,\tau)$ differs from $x'(k+1)$. An important quantity in our analysis is the size of $\|x'(k+1) - x(k+1)\|$. Next Lemma establishes a bound on this quantity.

Lemma 7.6 (Primal inexactness: Deterministic NJ) Consider Algorithm 8 under Assumptions 7.1 and 7.2. Then, for all $k = 0, 1, ...$:

$$\left\|x(k+1) - x'(k+1)\right\| \leq \left(\frac{\rho}{\rho + h_{\min}}\right)^\tau \|\widetilde{x}(k)\| + \left(\frac{\rho}{\rho + h_{\min}}\right)^\tau \frac{\|\widetilde{\eta}(k)\|}{h_{\min}}.$$

Proof: Note from (7.38) that $x'(k+1)$ obeys $\nabla F(x'(k+1)) + \rho\,\mathcal{L}\,x'(k+1) = -\eta(k)$. Hence, using $\mathcal{L} = I - W$ and the definition of Φ in (7.25):

$$x'(k+1) = \Phi^{-1}\left(\,\rho\, W x'(k+1) - \eta(k)\right). \tag{7.39}$$

Next, fix some s, $0 \leq s \leq \tau - 1$, and consider (7.36). Subtracting $x'(k+1)$ from both sides of (7.36), and using (7.39) and (7.32):

$$x(k, s+1) - x'(k+1) = R_\Phi(s)\,\rho\, W(x(k,s) - x'(k+1)),$$

where we introduce a simplified notation: $R_\Phi(s) := R_\Phi\left(\rho W x(k,s) - \eta(k),\ \rho W x'(k+1) - \eta(k)\right)$. Next,

using (7.34), and $\|W\| = 1$, obtain:

$$\|x(k, s+1) - x'(k+1)\| \leq \left(\frac{\rho}{\rho + h_{\min}}\right) \|x(k,s) - x'(k+1)\|.$$

Applying the latter successively for $s = 0, 1, ..., \tau - 1$, and using $x(k, \tau) = x(k+1)$, $x(k, 0) = x(k)$, obtain:

$$\begin{aligned} \|x(k+1) - x'(k+1)\| &\leq \left(\frac{\rho}{\rho + h_{\min}}\right)^{\tau} \|x(k) - x'(k+1)\| \\ &\leq \left(\frac{\rho}{\rho + h_{\min}}\right)^{\tau} \left(\|\widetilde{x}(k)\| + \|x^{\bullet} - x'(k+1)\|\right), \end{aligned} \tag{7.40}$$

where we used $\|x(k) - x'(k+1)\| = \|(x(k) - x^{\bullet}) + (x^{\bullet} - x'(k+1))\| \leq \|x(k) - x^{\bullet}\| + \|x^{\bullet} - x'(k+1)\| = \|\widetilde{x}(k)\| + \|x^{\bullet} - x'(k+1)\|$. We next upper bound $\|x'(k+1) - x^{\bullet}\|$. Note that $x^{\bullet} = \Psi^{-1}(-\eta^{\bullet})$. Using the latter, (7.38), and (7.33), we obtain:

$$x'(k+1) - x^{\bullet} = \Psi^{-1}(-\eta(k)) - \Psi^{-1}(\eta^{\bullet}) = -R_{\Psi}(k)\,(\eta(k) - \eta^{\bullet}), \tag{7.41}$$

with $R_{\Psi}(k) := R_{\Psi}(-\eta(k), -\eta^{\bullet})$. This, together with (7.35), and $\widetilde{\eta}(k) = \eta(k) - \eta^{\bullet}$, gives:

$$\|x'(k+1) - x^{\bullet}\| \leq \frac{1}{h_{\min}} \|\widetilde{\eta}(k)\|. \tag{7.42}$$

Substituting the latter in (7.40) completes the proof of the Lemma. □

We next upper bound the primal error $\|\widetilde{x}(k+1)\|$.

Lemma 7.7 (Primal error: Deterministic NJ) Let Assumptions 7.1 and 7.2 hold. Then, for all $k = 0, 1, ...$:

$$\|\widetilde{x}(k+1)\| \leq \left(\frac{\rho}{\rho + h_{\min}}\right)^{\tau} \|\widetilde{x}(k)\| + \frac{1}{h_{\min}} \left(1 + \left(\frac{\rho}{\rho + h_{\min}}\right)^{\tau}\right) \|\widetilde{\eta}(k)\|.$$

Proof: Write $\widetilde{x}(k+1) = (x(k+1) - x'(k+1)) + (x'(k+1) - x^{\bullet})$. Then, $\|\widetilde{x}(k+1)\| \leq \|x(k+1) - x'(k+1)\| + \|x'(k+1) - x^{\bullet}\|$. The result now follows by applying Lemma 7.6 and (7.42).

We proceed with bounding the dual error. For our final goal (bounding the primal error), rather than studying directly $\widetilde{\eta}(k) = \eta(k) - \eta^{\bullet}$, it is more useful to consider a certain transformed quantity. Represent the weighted Laplacian matrix $\mathcal{L}$ as $\mathcal{L} = Q\widehat{\Lambda}Q^{\top} = \sum_{i=2}^{N} \lambda_i\, q_i q_i^{\top}$. Here, λ_i is the i-th smallest eigenvalue of $\mathcal{L}$ ($\lambda_i > 0$, for all $i = 2, ..., N$); $Q = [q_2, ..., q_N]$ is the $N \times (N-1)$ matrix, and its column q_i is the unit-norm eigenvector of $\mathcal{L}$ that corresponds to λ_i; and $\widehat{\Lambda}$ is the $(N-1) \times (N-1)$ diagonal matrix with

diagonal $(\lambda_2, ..., \lambda_N)$. Next, introduce:

$$\widetilde{\eta}'(k) := Q^\top \widetilde{\eta}(k) \in \mathbb{R}^{N-1} \quad \text{and} \quad \widetilde{\eta}''(k) := \widehat{\Lambda}^{-1/2} \widetilde{\eta}'(k) \in \mathbb{R}^{N-1}. \tag{7.43}$$

We next bound the norm of $\widetilde{\eta}''(k+1)$.

Lemma 7.8 (Dual error: Deterministic NJ) Let Assumptions 7.1 and 7.2 hold, and suppose that $\alpha \leq h_{\min}$. Then, for all $k = 0, 1, ...$:

$$\|\widetilde{\eta}''(k+1)\| \leq \left[\left(1 - \frac{\alpha\, \lambda_2(\mathcal{L})}{h_{\max} + \rho} \right) + \frac{\alpha}{h_{\min}} \left(\frac{\rho}{\rho + h_{\min}} \right)^\tau \right] \|\widetilde{\eta}''(k)\| + \alpha \left(\frac{\rho}{\rho + h_{\min}} \right)^\tau \|\widetilde{x}(k)\|.$$

Consider (7.37). Because $\mathcal{L} x^\bullet = \mathcal{L} x^\star \, 1 = 0$, we have:

$$\mathcal{L} x(k+1) = \mathcal{L}(x(k+1) - x'(k+1)) + \mathcal{L}(x'(k+1) - x^\bullet).$$

Using the latter, and subtracting $\eta^\bullet$ from both sides of (7.37), obtain:

$$\widetilde{\eta}(k+1) = \widetilde{\eta}(k) + \alpha\, \mathcal{L}(x'(k+1) - x^\bullet) + \alpha\, \mathcal{L}(x(k+1) - x'(k+1)). \tag{7.44}$$

Further, using (7.41), we get:

$$\widetilde{\eta}(k+1) = (\, I - \alpha\, \mathcal{L}\, R_\Psi(k)\,)\, \widetilde{\eta}(k) + \alpha\, \mathcal{L}\, (x(k+1) - x'(k+1)). \tag{7.45}$$

Now, recall $\widetilde{\eta}'(k)$ in (7.43). It is easy to see that:

$$\|\widetilde{\eta}'(k)\| = \|\widetilde{\eta}(k)\|, \quad Q Q^\top \widetilde{\eta}(k) = \widetilde{\eta}(k). \tag{7.46}$$

Indeed, note that $1^\top \eta(k) = 1^\top \eta(k-1) + \alpha 1^\top \mathcal{L} x(k) = 1^\top \eta(k-1) = ... = 1^\top \eta(0) = 0$, because $\eta(0) = 0$ (by assumption.) Also, $1^\top \eta^\bullet = 0$ (see Lemma 7.5.) Therefore, $1^\top \widetilde{\eta}(k) = 0$, $\forall k$. Now, as $q_1 = \frac{1}{\sqrt{N}} 1$, we have $Q Q^\top \widetilde{\eta}(k) = \sum_{i=2}^N q_i q_i^\top \widetilde{\eta}(k) = \sum_{i=1}^N q_i q_i^\top \widetilde{\eta}(k) = \widetilde{\eta}(k)$; thus, the second equality in (7.46). For the first equality in (7.46), observe that: $\|\widetilde{\eta}'(k)\|^2 = (\widetilde{\eta}'(k))^\top \widetilde{\eta}'(k) = \widetilde{\eta}(k)^\top Q Q^\top \widetilde{\eta}(k) = \|\widetilde{\eta}(k)\|^2$.

Next, multiplying (7.45) from the left by $Q^\top$, expressing $\mathcal{L} = Q \widehat{\Lambda} Q^\top$, and using (7.46), obtain:

$$\widetilde{\eta}'(k+1) = \Big(I - \alpha\, \widehat{\Lambda}\, Q^\top R_\Psi(k)\, Q \Big)\, \eta'(k) + \alpha\, \widehat{\Lambda}\, Q^\top\, (x(k+1) - x'(k+1)). \tag{7.47}$$

Further, recall $\widetilde{\eta}''(k)$ in (7.43). Multiplying (7.47) from the left by $\widehat{\Lambda}^{-1/2}$, we obtain:

$$\widetilde{\eta}''(k+1) = \left(I - \alpha\, \widehat{\Lambda}^{1/2}\, Q^\top R_\Psi(k)\, Q \widehat{\Lambda}^{1/2} \right) \widetilde{\eta}''(k) + \alpha\, \widehat{\Lambda}^{1/2}\, Q^\top \left(x(k+1) - x'(k+1) \right). \tag{7.48}$$

Next, using variational characterizations of minimal and maximal eigenvalues, it is easy to verify that:

$$\frac{\lambda_2}{h_{\max} + \rho}\, I \preceq \widehat{\Lambda}^{1/2}\, Q^\top R_\Psi(k)\, Q \widehat{\Lambda}^{1/2} \preceq \frac{1}{h_{\min}}\, I. \tag{7.49}$$

By Assumption, $\alpha \leq h_{\min}$, and so:

$$\| I - \alpha\, \widehat{\Lambda}^{1/2}\, Q^\top R_\Psi(k)\, Q \widehat{\Lambda}^{1/2} \| \leq 1 - \frac{\alpha\, \lambda_2}{h_{\max} + \rho}. \tag{7.50}$$

Using the latter, $\|\widehat{\Lambda}^{1/2}\| \leq 1$ (as $0 \preceq \mathcal{L} \preceq I$), $\|Q\| = 1$, and Lemma 7.6, we get:

$$\|\widetilde{\eta}''(k+1)\| \leq \left(1 - \frac{\alpha\, \lambda_2}{h_{\max} + \rho} \right) \|\widetilde{\eta}''(k)\| + \alpha \left(\frac{\rho}{\rho + h_{\min}} \right)^\tau \|\widetilde{x}(k)\| + \alpha \left(\frac{\rho}{\rho + h_{\min}} \right)^\tau \frac{\|\widetilde{\eta}(k)\|}{h_{\min}}. \tag{7.51}$$

Finally, using $\|\widetilde{\eta}(k)\| = \|\widetilde{\eta}'(k)\| = \|\widehat{\Lambda}^{1/2} \widetilde{\eta}''(k)\| \leq \|\widetilde{\eta}''(k)\|$, we obtain the desired result. □

We are now ready to prove Theorem 7.3.

[Proof of Theorem 7.3] Introduce $\nu(k) := \frac{2}{h_{\min}} \|\widetilde{\eta}(k)\|$. Further, denote by $c_{11} := \left(\frac{\rho}{\rho + h_{\min}} \right)^\tau$, $c_{12} := \frac{1}{2} \left[1 + \left(\frac{\rho}{\rho + h_{\min}} \right)^\tau \right]$; $c_{21} := \frac{2\,\alpha}{h_{\min}} \left(\frac{\rho}{\rho + h_{\min}} \right)^\tau$, and $c_{22} := \left(1 - \frac{\alpha\, \lambda_2}{h_{\max} + \rho} \right) + \frac{\alpha}{h_{\min}} \left(\frac{\rho}{\rho + h_{\min}} \right)^\tau$. Using $\|\widetilde{\eta}(k)\| \leq \|\widetilde{\eta}''(k)\|$, Lemma 7.7 and Lemma 7.8, we obtain:

$$\max \left\{ \|\widetilde{x}(k+1)\|,\, \nu(k+1) \right\} \leq r\, \max \left\{ \|\widetilde{x}(k)\|,\, \nu(k) \right\},$$

with $r = \max \left\{ c_{11} + c_{12},\, c_{21} + c_{22} \right\}$. Unwinding the above recursion, using:

$$\|\widetilde{x}(k)\| \leq \max \{ \|\widetilde{x}(k)\|,\, \nu(k) \},$$

and using $\nu(0) = \frac{2}{h_{\min}} \|\widehat{\Lambda}^{-1/2} Q^\top \widetilde{\eta}(0)\| = \frac{2}{h_{\min}} \|\widehat{\Lambda}^{-1/2}\, Q^\top \left(-\nabla F(x^\star\, 1) \right)\| \leq \frac{2}{h_{\min} \sqrt{\lambda_2}} \sqrt{N} D_\eta$, we obtain (7.20).

It remains to show that that $r < 1$ if conditions (7.11) and (7.12) hold. Note that:

$$c_{11} + c_{12} = \frac{1}{2} + \frac{3}{2} \left(\frac{\rho}{\rho + h_{\min}} \right)^\tau,$$

and so $c_{11} + c_{12} < 1$ if:

$$\left(\frac{\rho}{\rho + h_{\min}}\right)^{\tau} < \frac{1}{3}. \tag{7.52}$$

Next, note that:

$$c_{21} + c_{22} = \left(1 - \frac{\alpha\,\lambda_2}{\rho + h_{\max}}\right) + \frac{3\alpha}{h_{\min}}\left(\frac{\rho}{\rho + h_{\min}}\right)^{\tau},$$

and so $c_{21} + c_{22} < 1$ if:

$$\left(\frac{\rho}{\rho + h_{\min}}\right)^{\tau} < \frac{1}{3}\left(\frac{h_{\min}\lambda_2}{\rho + h_{\max}}\right). \tag{7.53}$$

Combining (7.52) and (7.53), we obtain that $r < 1$ if conditions (7.11) and (7.12) hold. The proof is complete. □

7.3.3 Auxiliary lemmas and proof of Theorem 7.4

We now perform the convergence analysis of the deterministic algorithm variant with gradient primal variable updates (Algorithm 7 where (7.7) is replaced with (7.10)), and our goal is to prove Theorem 7.4. The difference with respect to deterministic nonlinear Jacobi variant is only in the primal (inner) updates. Hence, much of the analysis in Subsection 7.3.2 continues to hold here. The key difference is in Lemma 7.6. We now state and prove a counterpart of Lemma 7.6 with the gradient primal updates.

Lemma 7.9 (Primal inexactness: Deterministic gradient updates) Consider Algorithm 7 where step (7.7) is replaced with (7.10), and let Assumptions 7.1 and 7.2 hold. Further, suppose that $\beta \leq \frac{1}{\rho + h_{\max}}$. Then, for all $k = 0, 1, ...$:

$$\left\|x(k+1) - x'(k+1)\right\| \leq (1 - \beta\, h_{\min})^{\tau}\, \|\widetilde{x}(k)\| + (1 - \beta\, h_{\min})^{\tau}\, \frac{\|\widetilde{\eta}(k)\|}{h_{\min}}.$$

With Lemma 7.9 in force, the rest of the analysis follows from the analysis in Subsection 7.3.2. It can be easily verified that all the results (Lemma 7.7, Lemma 7.8, and Theorem 7.3) continue to hold, with the quantity $\frac{\rho}{\rho + h_{\min}}$ replaced with $1 - \beta\, h_{\min}$. Henceforth, our task of proving Theorem 7.4 is completed once we prove Lemma 7.9. Introduce again the same compact notation $x(k), x(k,s)$, and $\eta(k)$, as with the deterministic NJ variant. We also make use of the quantity $x'(k+1)$ in (7.38). Finally, use the same notation – $\widetilde{x}(k)$ and $\widetilde{\eta}(k)$ – for the primal and dual errors.

Proof: In compact notation, using $\mathcal{L} = I - W$, the update (7.10) is rewritten as:

$$x(k, s+1) = x(k,s) - \beta\, \left(\rho\, \mathcal{L}\, x(k,s) + \eta(k) + \nabla F(x(k,s))\right). \tag{7.54}$$

This is precisely the gradient descent on $L_a(\cdot;\eta(k))$ in (7.3). Further, recall $x'(k+1)$ in (7.38). As $x'(k+1)$ satisfies $\rho\,\mathcal{L}\,x'(k+1)+\eta(k)+\nabla F(x(k+1))=0$, we have:

$$x'(k+1)=x'(k+1)-\beta\,\left(\rho\,\mathcal{L}\,x'(k+1)+\eta(k)+\nabla F(x'(k+1))\right). \tag{7.55}$$

Further, by Assumption 7.1, we have that $\nabla F:\,\mathbb{R}^N\to\mathbb{R}^N$ is continuously differentiable, and there holds:

$$\begin{aligned}\nabla F(x(k,s))-\nabla F(x'(k+1)) &= \left[\int_{z=0}^{1}\nabla^2 F\left(x'(k+1)+z(x(k,s)-x'(k+1))\right)dz\right]\\ &\times\quad (x(k,s)-x'(k+1))\\ &=:\quad H_F\,(s)\;(x(k,s)-x'(k+1)).\end{aligned} \tag{7.56}$$

Further, by Assumption 7.1, the matrix $H_F\,(s)$ satisfies:

$$h_{\min}\,I\preceq H_F(s)\preceq h_{\max}\,I. \tag{7.57}$$

Using (7.56), and subtracting (7.55) from (7.54), we obtain:

$$x(k,s+1)-x'(k+1)=(I-\beta\,\rho\,\mathcal{L}-\beta\,H_F(s))\;(x(k,s)-x'(k+1)). \tag{7.58}$$

Consider the matrix $(I-\beta\,\rho\,\mathcal{L}-\beta\,H_F(s))$. As $\beta\leq\frac{1}{\rho+h_{\max}}$ (by assumption), using (7.57) and $0\preceq\mathcal{L}\preceq I$, we have that: $(I-\beta\,\rho\,\mathcal{L}-\beta\,H_F(s))\succeq 0$. Thus, $\|I-\beta\,\rho\,\mathcal{L}-\beta\,H_F(s)\|\leq 1-\lambda_1\,(\beta\,\rho\,\mathcal{L}+\beta\,H_F(s))\leq 1-\beta\,h_{\min}$. Applying the latter bound to (7.58), we obtain the following important inequality:

$$\|x(k,s+1)-x'(k+1)\|\leq(1-\beta\,h_{\min})\;\|x(k,s)-x'(k+1)\|. \tag{7.59}$$

Applying (7.59) successively for $s=0,...,\tau-1$, using $x(k,s=0)=x(k)$, and $x(k,s=\tau)=x(k+1)$, we get:

$$\|x(k+1)-x'(k+1)\|\leq(1-\beta\,h_{\min})^{\tau}\;\|x(k)-x'(k+1)\|. \tag{7.60}$$

The proof now proceeds analogously to the proof of Lemma 7.6. □

7.4 Randomized Distributed Augmented Lagrangian Methods

This section studies randomized distributed AL methods. Subsection 7.4.1 explains the asynchronous communication model and the two algorithm variants (with the NGS updates and the gradient updates.) Subsection 7.4.2 states the results on their convergence rates and gives interpretations.

7.4.1 Model and algorithms

We continue to consider: 1) the optimization model defined by Assumption 7.1; and 2) a N-node connected network $\mathcal{G}$ (Assumption 7.2 holds), with the associated matrices W and $\mathcal{L}$. We consider two variants of the randomized distributed AL methods of type (7.1)–(7.2). Both utilize the same communication protocol, but they mutually differ in the way primal variables are updated. Like the deterministic counterparts, they both update the dual variables at the outer iterations k, and they update the primal variables at the inner iterations s. At each inner iteration s, one node, say i, is selected uniformly at random from the set of nodes $\{1, 2, ..., N\}$. Upon selection, node i updates its primal variable and broadcasts it to all its neighbors. We now detail the time and communication models. The outer iterations occur at discrete time steps of the physical time; k-th outer iteration occurs at time $\tau\, k$, $k = 1, 2, ...$, i.e., every τ time units. We assume that all nodes have synchronized clocks for the dual variable updates (dual variable clocks). Each node i has another clock (primal variable clock) that ticks according to a rate 1 Poisson process; on average, there is one tick of node i in the time interval of width 1. Whenever node i's Poisson clock ticks, node i updates its primal variable and broadcasts it to neighbors. Further, the Poisson process clocks of different nodes are independent. Consider the Poisson process clock that ticks whenever one of the nodes' clocks ticks. This process is a rate-N Poisson process. Hence, in the time interval of length τ, there are, on average, $\tau\, N$ ticks (primal updates), out of which τ (on average) are done by i.[4]

More formally, let $(\Theta, \mathcal{F}, \mathbb{P})$ be a probability space. Further, let $\{\mathcal{T}_i(a, b]\}_{0 \leq a \leq b < \infty}$ be a Poisson process with rate 1, $i = 1, ..., N$. (This is the node i's clock for primal variables.) Thus, for a fixed a, b, $\mathcal{T}_i(a, b] : \Theta \to \mathbb{R}$, $\mathcal{T}_i(a, b] = \mathcal{T}_i((a, b]\,; \omega)$, $\omega \in \Theta$, is a Poisson random variable with mean $(b - a)$. We assume that the processes $\mathcal{T}_i$ are independent. Further, let $\mathcal{T}$ be a Poisson process defined by $\mathcal{T}(a, b] := \sum_{i=1}^{N} \mathcal{T}_i(a, b]$. Define the random variable $\tau(k) := \mathcal{T}(k\tau, (k+1)\tau]$ (the number of ticks across all nodes in the k-the outer iteration.) Next, consider the events $\mathcal{A}_{k,j}$, $j = 0, 1, 2, ...$, defined by $\mathcal{A}_{k,j} := \{\omega \in \Theta : \tau(k; \omega) = j\}$. For $j \geq 1$, we also define the maps: $\widehat{\imath}(k, s) : \mathcal{A}_{k,j} \to \{1, 2, ..., N\}$, $s = 0, ..., j - 1$, as follows: $\widehat{\imath}(k, s; \omega) = i$, if the $(s + 1)$-th tick of $\mathcal{T}$ in the interval $(k\tau, (k + 1)\tau]$ comes from node i's clock $\mathcal{T}_i$.

[4] Note that one primal update here corresponds to an update of a *single* node. Thus, roughly, N updates (ticks) here correspond to one update (inner) iteration of the deterministic algorithm.

We now present the two variants of the randomized distributed AL algorithm. The first updates the primal variables via a NGS method; the alternative variant replaces the NGS updates by the gradient type updates. We next detail the NGS variant.

NGS updates. We observe the primal and dual variables at times $k\tau$, $k = 0, 1, ...$ These are the time instances when the dual variables are updated (We assume that the dual variables are updated instantaneously at the moments $k\tau$.) We denote by $x_i(k) := x_i(k\tau)$ the node i's primal variable at time $k\tau$, $k = 0, 1, ...$ Further, consider $\omega \in \mathcal{A}_{k,j}$: the total number of ticks $\tau(k)$ of $\mathcal{T}$ in the interval $(k\tau, (k+1)\tau]$ equals j, and hence we have j inner iterations (ticks) at the outer iteration k. For any $\omega \in \mathcal{A}_{k,j}$, we denote by $x_i(k, s)$ the node i's variable after the s-th inner iteration, $s = 1, ..., j$, $j \geq 1$. Also, we denote by $x_i(k, 0) := x_i(k)$, and, for $\omega \in \mathcal{A}_{k,j}$, $x_i(k, \tau(k) = j) := x_i(k+1)$. Each node maintains: 1) the primal variable $x_i(k)$; 2) the dual variable $\eta_i(k) := \eta_i(k\tau)$; the (weighted) sum of the neighbors' variables $\overline{x}_i(k) := \sum_{j \in O_i} W_{ij} x_j(k)$, as well as the analogous intermediate variables $x_i(k, s)$ and $\overline{x}_i(k, s)$ during the inner iterations s. The algorithm is Summarized in Algorithm 9.

Algorithm 9 Randomized distributed AL algorithm with NGS updates

1: (**Initialization**) Node i sets $k = 0$, $x_i(k = 0) \in \mathbb{R}^d$, $\overline{x}_i(k = 0) = x_i(k = 0)$, and $\eta_i(k = 0) = 0$.

2: (**Inner iterations**) Set $x_i(k, s = 0) := x_i(k)$, $\overline{x}_i(k, s = 0) := \overline{x}_i(k)$, and $s = 0$. If $\omega \in \Theta$ is such that $\tau(k) = \tau(k; \omega) > 0$, then, for $s = 0, 1, ..., \tau(k) - 1$, do (else, if $\tau(k; \omega) = 0$, then go to step 3):

Update the inner variables $x_j(k, s)$, $j = 1, ..., N$, by :

$$x_j(k, s+1) = \begin{cases} \arg\min_{x_j \in \mathbb{R}^d} \left(f_j(x_j) + (\eta_j(k) - \rho\, \overline{x}_j(k, s))^\top x_j + \frac{\rho \|x_j\|^2}{2} \right) & \text{for } j = \widehat{\imath}(k, s) \\ x_j(k, s+1) = x_j(k, s) & \text{else.} \end{cases} \tag{7.61}$$

Update the variables $\overline{x}_j(k, s)$, $j = 1, ..., N$, by :

$$\overline{x}_j(k, s+1) = \begin{cases} \sum_{l \in \Omega_j} W_{jl}\, x_l(k, s+1) & \text{for } j \in O_i \ : i = \widehat{\imath}(k, s) \\ \overline{x}_j(k, s+1) = \overline{x}_j(k, s) & \text{else;} \end{cases} \tag{7.62}$$

and all nodes $j = 1, ..., N$ set $x_j(k+1) := x_j(k, s = \tau(k))$, $\overline{x}_j(k+1) = x_j(k, s = \tau(k))$.

3: (**Outer iteration**) All nodes j update the dual variables $\eta_j(k)$ via:

$$\eta_j(k+1) = \eta_j(k) + \alpha\, (x_j(k+1) - \overline{x}_j(k+1)). \tag{7.63}$$

4: Set $k \mapsto k+1$ and go to step 2.

Gradient primal updates. This algorithm variant is the same as given in Algorithm 9, except that step (7.61) is replaced with the following:

$$x_j(k, s+1) = \begin{cases} (1 - \beta\, \rho)\, x_j(k, s) + \beta\, \rho\, \overline{x}_j(k, s) - \beta\, (\eta_j(k) + \nabla f_j(x_j(k, s))) & \text{for } j = \widehat{\imath}(k, s) \\ x_j(k, s+1) = x_j(k, s) & \text{else.} \end{cases} \tag{7.64}$$

Here, $\beta > 0$ is the (primal) step-size parameter.

7.4.2 Convergence rate: Statements of results

We now state the convergence rate result for the randomized AL method with NGS updates.

Theorem 7.10 (Convergence rate: Randomized NGS) Consider Algorithm 9 under Assumptions 7.1 and 7.2, and suppose that the algorithm and network parameters satisfy the following:

$$\alpha \leq h_{\min} \tag{7.65}$$

$$e^{-\sigma\,\tau} < \frac{1}{3}\,\frac{\lambda_2(\mathcal{L})\,h_{\min}}{\rho + h_{\max}}, \tag{7.66}$$

where

$$\sigma := N\left\{1 - \left[1 - \frac{1}{N}\left(1 - \frac{\rho^2}{(\rho + h_{\min})^2}\right)\right]^{1/2}\right\}. \tag{7.67}$$

Then, at any node i, $\mathbb{E}\,[\,\|x_i(k) - x^\star\|\,]$ generated by Algorithm 9 converges linearly (in the outer iterations k) to zero, with rate:

$$r_{\text{rand,ngs}} := \max\left\{\frac{1}{2} + \frac{3}{2}\,e^{-\sigma\,\tau},\ \left(1 - \frac{\alpha\,\lambda_2(\mathcal{L})}{\rho + h_{\max}}\right) + \frac{3\alpha}{h_{\min}}\,e^{-\sigma\,\tau}\right\} < 1, \tag{7.68}$$

and there holds:

$$\mathbb{E}\,[\,\|x_i(k) - x^\star\|\,] \leq (r_{\text{rand,ngs}})^k\,\sqrt{N}\,\max\left\{D_x,\ \frac{2\,D_\eta}{\sqrt{\lambda_2(\mathcal{L})\,h_{\min}}}\right\}. \tag{7.69}$$

For a large N, σ is approximated as:

$$\sigma \approx \frac{1}{2}\left(1 - \frac{\rho^2}{(\rho + h_{\min})^2}\right)^2.$$

Condition (7.66) is satisfied, for example, if $\rho \leq h_{\min}$, and:

$$\tau \geq \left\lceil \frac{\left|\log\left(\frac{3(1+\gamma)}{\lambda_2(\mathcal{L})}\right)\right|}{N\,\left(1 - (1 - 3/(4\,N))^{1/2}\right)} \right\rceil. \tag{7.70}$$

The communication rate in Table 7.1 is obtained by taking:

$$\alpha = \rho = h_{\min},\quad \beta = \frac{1}{\rho + h_{\max}},\quad \tau = \left\lceil \frac{\log\left(\frac{6(1+\gamma)}{\lambda_2(\mathcal{L})}\right)}{N\,\left(1 - (1 - 3/(4\,N))^{1/2}\right)} \right\rceil.$$

We now present a similar result for the randomized AL method with gradient-type updates.

Theorem 7.11 (Convergence rate: Randomized gradient updates) Consider Algorithm 9 where step (7.61) is replaced with (7.64), and let Assumptions 7.1 and 7.2 hold. Further, suppose that the algorithm and network parameters satisfy the following:

$$\alpha \leq h_{\min} \tag{7.71}$$

$$\beta \leq \frac{1}{\rho + h_{\max}} \tag{7.72}$$

$$e^{-\sigma' \tau} < \frac{1}{3} \frac{\lambda_2 \, h_{\min}}{\rho + h_{\max}}. \tag{7.73}$$

where

$$\sigma' := N \left\{ 1 - \left[1 - \frac{1}{N} \beta \, h_{\min} (1 - \beta \, h_{\min}) \right]^{1/2} \right\}. \tag{7.74}$$

Then, at any node i, $\mathbb{E}\,[\,\|x_i(k) - x^\star\|\,]$ generated by Algorithm 9 converges linearly (in the outer iterations k) to zero, with rate:

$$r_{\text{rand,grad}} := \max \left\{ \frac{1}{2} + \frac{3}{2} \, e^{-\sigma' \tau}, \; \left(1 - \frac{\alpha \, \lambda_2(\mathcal{L})}{\rho + h_{\max}} \right) + \frac{3\alpha}{h_{\min}} \, e^{-\sigma' \tau} \right\} < 1, \tag{7.75}$$

and there holds:

$$\mathbb{E}\,[\,\|x_i(k) - x^\star\|\,] \leq (r_{\text{rand,grad}})^k \, \sqrt{N} \, \max \left\{ D_x, \; \frac{2 \, D_\eta}{\sqrt{\lambda_2(\mathcal{L})} \, h_{\min}} \right\}. \tag{7.76}$$

For a large N, σ' is approximated as:

$$\sigma' \approx \frac{1}{2} \, \beta h_{\min} (1 - \beta \, h_{\min}).$$

Condition (7.73) is satisfied, for example, if $\rho \leq h_{\min}$, $\beta = \frac{1}{\rho + h_{\max}}$, and:

$$\tau \geq \left\lceil \frac{\left| \log \left(\frac{3(1+\gamma)}{\lambda_2(\mathcal{L})} \right) \right|}{N \left(1 - \left(1 - \frac{\gamma}{N(1+\gamma)^2} \right)^{1/2} \right)} \right\rceil. \tag{7.77}$$

We can see that, with respect to the randomized NGS variant, the τ that ensures linear convergence grows faster with γ. The communication rate in Table 7.1 is obtained with τ twice larger than in (7.77), $\alpha = \rho = h_{\min}$, and $\beta = \frac{1}{\rho + h_{\max}}$.

7.5 Convergence rate analysis: Proofs of Theorems 7.10 and 7.11

The goal of this Section is to prove Theorems 7.10 and 7.11. Subsection 7.5.1 sets up the analysis by introducing certain maps akin to the map Φ in Section 7.3. Subsection 7.5.2 establishes the desired results.

7.5.1 Setting up analysis

For sake of a clean notation, as in Section 7.3, we assume that $d = 1$, but the analysis extends to a generic $d > 1$ as well. For each $i = 1, ..., N$, we introduce the following map: $\Phi_i : \mathbb{R} \mapsto \mathbb{R}$:

$$\Phi_i(x) \quad := \quad \nabla f_i(x) + \rho\, x. \tag{7.78}$$

Next, define the map: $\Phi_i^{-1} : \mathbb{R} \to \mathbb{R}$ by:

$$\Phi_i^{-1}(\eta) \quad := \quad \arg\min_{y \in \mathbb{R}} \left(f_i(y) - \eta_i\, y + \frac{\rho}{2}\, y^2 \right). \tag{7.79}$$

For any $\eta \in \mathbb{R}$, the above map is well-defined by Assumption 7.1 (The cost in (7.79) is strongly convex.) Similarly to Subsection 7.3.1, it can be shown that: 1) Φ^{-1} is the inverse of Φ; 2) $\Phi^{-1} : \mathbb{R} \to \mathbb{R}$ is continuously differentiable; and 3) the derivative is:

$$\nabla \Phi_i^{-1}(\eta) \quad = \quad \left(\nabla^2 f_i(\Phi_i^{-1}(\eta)) + \rho \right)^{-1}. \tag{7.80}$$

Again, similarly to Subsection 7.3.1, it can be shown that:

$$\Phi_i^{-1}(\eta_1) - \Phi_i^{-1}(\eta_2) \quad = \quad R_{\Phi,i}(\eta_1, \eta_2)\, (\eta_1 - \eta_2) \tag{7.81}$$

$$R_{\Phi,i}(\eta_1, \eta_2) \quad := \quad \int_{z=0}^{1} \nabla \Phi_i^{-1}(\eta_1 + z(\eta_2 - \eta_1))\, dz \quad (\in \mathbb{R}) \tag{7.82}$$

$$\frac{1}{h_{\max} + \rho} \quad \leq \quad R_{\Phi,i}(\eta_1, \eta_2) \leq \frac{1}{h_{\min} + \rho}, \quad \forall \eta_1, \eta_2 \in \mathbb{R}. \tag{7.83}$$

7.5.2 Auxiliary Lemmas and proofs of Theorems 7.10 and 7.11

We mimic the structure of Subsection 7.3.2. We first establish the primal inexactness bound, proceed with primal and dual error bounds, and finalize by proving Theorems 7.10 and 7.11. When the proofs are similar to that of the already established results, we curtail repetitive arguments. Consider $x'(k+1)$ in (7.38).

Lemma 7.12 (Primal inexactness: Randomized NGS) Consider Algorithm 9 under Assumptions 7.1 and 7.2.

Then, for all $k = 0, 1, ...$:

$$\mathbb{E}\left[\left\|x(k+1) - x'(k+1)\right\|\right] \leq e^{-\sigma\,\tau}\,\mathbb{E}\left[\|\widetilde{x}(k)\|\right] + e^{-\sigma\,\tau}\,\frac{1}{h_{\min}}\,\mathbb{E}\left[\|\widetilde{\eta}(k)\|\right],$$

where σ is given in (7.67).

Proof: Fix some k, fix some $j = 1, 2, ...$, and take $\omega \in \mathcal{A}_{k,j}$. Thus, $\tau(k) = \tau(k;\omega) = j$ and there are j inner iterations. Fix some s, $s \in \{0, 1, ..., j-1\}$, and suppose that $\widehat{\imath}(k,s) = i$ (node i is activated.) We have that $x_i(k, s+1)$ satisfies the following:

$$x_i(k, s+1) = \Phi_i^{-1}\left(\rho \sum_{j \in O_i} W_{ij}\, x_j(k,s) - \eta_i(k)\right).$$

On the other hand, we know that $x_i'(k+1)$ satisfies:

$$x_i'(k+1) = \Phi_i^{-1}\left(\rho \sum_{j \in O_i} W_{ij}\, x_j'(k+1) - \eta_i(k)\right).$$

Subtracting the above equalities, and using (7.82)–(7.83), letting

$$R_{\Phi,i}(s) := R_{\Phi,i}\left(\rho \sum_{j \in O_i} W_{ij}\, x_j(k,s) - \eta_i(k)\,,\, \rho \sum_{j \in O_i} W_{ij}\, x_j'(k+1) - \eta_i(k)\right),$$

and squaring the equality, we obtain:

$$\begin{aligned}
\big(x_i(k, s+1) - x_i'(k+1)\big)^2 &= (R_{\Phi,i}(s))^2\,\rho^2 \left(\sum_{j \in O_i} W_{ij}\,(x_j(k,s) - x_j'(k+1))\right)^2 \\
&\leq \left(\frac{\rho}{\rho + h_{\min}}\right)^2 \sum_{j \in O_i} W_{ij}\,(x_j(k,s) - x_j'(k+1))^2 \qquad (7.84) \\
&= \delta^2 \sum_{j=1}^{N} W_{ij}\;(x_j(k,s) - x_j'(k+1))^2. \qquad (7.85)
\end{aligned}$$

Here, (7.84) further uses: 1) convexity of the quadratic function $u \mapsto u^2$; 2) the fact that $\sum_{j \in O_i} W_{ij} = 1$; and 3) the fact that the W_{ij}'s are nonnegative. Also, (7.85) introduces notation: $\delta := \frac{\rho}{\rho + h_{\min}}$, and uses the fact that $W_{ij} = 0$ if $\{i,j\} \notin E$ and $i \neq j$. As node i is selected, the remaining quantities $x_j(k,s)$, $j \neq i$, remain unchanged; i.e., $x_j(k,s+1) - x_j'(k+1) = x_j(k,s) - x_j'(k+1)$, $j \neq i$. Squaring the latter equalities,

adding them up for all $j \neq i$, and finally adding them to (7.85), we obtain:

$$\begin{aligned} \|x(k,s+1) - x'(k+1)\|^2 &\leq \|x(k,s) - x'(k)\|^2 + \delta^2 \sum_{j=1}^{N} W_{ij} \left(x_j(k,s) - x'_j(k+1)\right)^2 \\ &- \left(x_i(k,s) - x'_i(k+1)\right)^2, \end{aligned} \tag{7.86}$$

for any $\omega \in \mathcal{A}_{k,j}$ such that $\widehat{\imath}(k,s) = i$.

We now compute conditional expectation of $\|x(k,s+1) - x'(k+1)\|^2$, conditioned on $\tau(k) = j$, $x(k) = x(k,0)$, $\eta(k)$, and $x(k,1), ..., x(k,s)$, $s \leq j-1$. Conditioned on the latter random variables, each node i updates equally likely, with conditional probability $1/N$, and therefore:

$$\mathbb{E}\left[\|x(k,s+1) - x'(k+1)\|^2 \mid x(k), \eta(k), \tau(k) = j, x(k,1), ..., x(k,s) \right] \tag{7.87}$$

$$\leq \|x(k,s) - x'(k+1)\|^2 + \frac{1}{N}\,\delta^2 \sum_{i=1}^{N} \sum_{j=1}^{N} W_{ij} \left(x_j(k,s) - x'_j(k+1)\right)^2 \tag{7.88}$$

$$- \frac{1}{N} \sum_{i=1}^{N} (x_i(k,s) - x'_i(k+1))^2 \tag{7.89}$$

$$= \|x(k,s) - x'(k+1)\|^2 + \frac{1}{N}\,\delta^2 \sum_{i=1}^{N} W_{ij} \sum_{j=1}^{N} \left(x_j(k,s) - x'_j(k+1)\right)^2$$

$$- \frac{1}{N} \|x(k,s) - x'(k+1)\|^2 \tag{7.90}$$

$$= \|x(k,s) - x'(k+1)\|^2 + \frac{1}{N}\,\delta^2 \|x(k,s) - x'(k+1)\|^2$$

$$- \frac{1}{N} \|x(k,s) - x'(k+1)\|^2, \; \forall \omega \in \mathcal{A}_{k,j}. \tag{7.91}$$

Here, inequality (7.91) uses the fact that $\sum_{i=1}^{N} W_{ij} = 1, \forall j$. Rewriting (7.91), we get:

$$\begin{aligned} &\mathbb{E}\left[\|x(k,s+1) - x'(k+1)\|^2 \mid x(k), \eta(k), \tau(k) = j, x(k,1), ..., x(k,s-1) \right] \\ &\leq \left(1 - \frac{1}{N}(1-\delta^2)\right) \left\|x(k,s) - x'(k+1)\right\|^2, \; \forall \omega \in \mathcal{A}_{k,j}. \end{aligned}$$

Denote by $\delta' := \left(1 - \frac{1}{N}(1-\delta^2)\right)^{1/2}$. Using the Jensen inequality for quadratic convex functions, we obtain:

$$\begin{aligned} &\mathbb{E}\left[\|x(k,s+1) - x'(k+1)\| \mid x(k), \eta(k), \tau(k) = j, x(k,1), ..., x(k,s-1) \right] \\ &\leq \delta' \left\|x(k,s) - x'(k+1)\right\|, \; \forall \omega \in \mathcal{A}_{k,j}. \end{aligned}$$

Integrating with respect to $x(k,1),...,x(k,s)$:

$$\mathbb{E}\left[\,\|x(k,s+1)-x'(k+1)\|\,|\,x(k),\eta(k),\tau(k)=j\,\right] \\ \leq \delta'\,\mathbb{E}\left[\left\|x(k,s)-x'(k+1)\right\|\,|\,x(k),\eta(k),\tau(k)=j\right],\ \forall\omega\in\mathcal{A}_{k,j}.$$

Applying the above inequality for $s=0,1,...,j-1$, using $x(k,s=\tau(k)=j)=x(k+1)$, and considering all the values of $j=0,1,...$, we obtain:

$$\mathbb{E}\left[\,\|x(k+1)-x'(k+1)\|\,|\,x(k),\eta(k),\tau(k)\,\right] \\ \leq (\delta')^{\tau(k)}\,\mathbb{E}\left[\left\|x(k)-x'(k+1)\right\|\,|\,x(k),\eta(k),\tau(k)\right],\ \text{almost surely (a.s.)}$$

Integrating with respect to $x(k),\eta(k)$:

$$\mathbb{E}\left[\,\|x(k+1)-x'(k+1)\|\,|\,\tau(k)=j\,\right] \\ \leq (\delta')^{j}\,\mathbb{E}\left[\left\|x(k)-x'(k+1)\right\|\,|\,\tau(k)=j\right]=(\delta')^{\tau(k)}\,\mathbb{E}\left[\left\|x(k)-x'(k+1)\right\|\,\right],\ \text{a.s.},$$

where we used independence of $\tau(k)$ and $x(k),\eta(k)$. Taking expectation, we obtain:

$$\mathbb{E}\left[\,\|x(k+1)-x'(k+1)\|\right] \\ \leq \mathbb{E}[\,(\delta')^{\tau(k)}\,]\;\mathbb{E}\left[\left\|x(k)-x'(k+1)\right\|\right].$$

Because $\tau(k)$ is distributed according to the Poisson distribution with parameter $N\,\tau$, we have: $\mathbb{E}\left[(\delta')^{\tau(k)}\right]=\sum_{l=0}^{\infty}(\delta')^{l}\,\frac{e^{-N\tau}(N\tau)^{l}}{l}=e^{-(1-\delta')N\,\tau}$. We get:

$$\mathbb{E}\left[\,\|x(k+1)-x'(k+1)\|\,\right]\leq\;e^{-(1-\delta')N\,\tau}\,\mathbb{E}\left[\left\|x(k)-x'(k+1)\right\|\right]. \tag{7.92}$$

Next, we use $x(k)-x'(k+1)=(x(k)-x^{\bullet})+(x^{\bullet}-x'(k+1))=\widetilde{x}(k)+(x^{\bullet}-x'(k+1))$, and so $\|x(k)-x'(k+1)\|\leq\|\widetilde{x}(k)\|+\|x^{\bullet}-x'(k+1)\|$. Further, we use the relation in (7.41), which gives $\|x'(k+1)-x^{\bullet}\|\leq\frac{1}{h_{\min}}\|\widetilde{\eta}(k)\|$. Thus, $\mathbb{E}\left[\|x(k)-x'(k+1)\|\right]\leq\mathbb{E}[\|\widetilde{x}(k)\|]+\frac{1}{h_{\min}}\,\mathbb{E}[\|\widetilde{\eta}(k)\|]$. Plugging the latter inequality in (7.92), and substituting $\delta'=\left(1-\frac{1}{N}\left(1-\frac{\rho^2}{(\rho+h_{\min})^2}\right)\right)^{1/2}$, we obtain the desired result. We now state and prove a Lemma on the primal error $\widetilde{x}(k+1)$.

Lemma 7.13 (Primal error: Randomized NGS) Consider Algorithm 9 under Assumptions 7.1 and 7.2. Then,

for all $k = 0, 1, ...$:

$$\mathbb{E}\,[\,\|\widetilde{x}(k+1)\|\,] \leq e^{-\sigma\,\tau}\,\mathbb{E}\,[\,\|\widetilde{x}(k)\|\,] + \frac{1}{h_{\min}} e^{-\sigma\,\tau}\,\mathbb{E}\,[\,\|\widetilde{\eta}(k)\|\,],$$

where σ is given in (7.67).

We have that $\widetilde{x}(k+1) = x(k+1) - x^{\bullet} = (x(k+1) - x'(k+1)) + (x'(k+1) - x^{\bullet})$, a.s., and so: $\|\widetilde{x}(k+1)\| \leq \|x(k+1) - x'(k+1)\| + \|x'(k+1) - x^{\bullet}\|$, a.s. Further, by (7.41), we have that $\|x'(k+1) - x^{\bullet}\| \leq \frac{1}{h_{\min}} \|\widetilde{\eta}(k)\|$, a.s. The Lemma now follows by taking expectation and applying Lemma 7.12. Introduce the (random) transformed dual error $\widetilde{\eta}''(k)$ as in (7.43).

Lemma 7.14 (Dual error: Randomized NGS) Consider Algorithm 9 under Assumptions 7.1 and 7.2, and suppose that $\alpha \leq h_{\min}$. Then, for all $k = 0, 1, ...$:

$$\mathbb{E}\,[\,\|\widetilde{\eta}''(k+1)\|\,] \leq \left[\,\left(1 - \frac{\alpha\,\lambda_2(\mathcal{L})}{h_{\max} + \rho}\right) + \frac{\alpha}{h_{\min}}\,e^{-\sigma\,\tau}\,\right]\,\mathbb{E}\,[\,\|\widetilde{\eta}''(k)\|\,] + \alpha\,e^{-\sigma\,\tau}\,\mathbb{E}\,[\,\|\widetilde{x}(k)\|\,],$$

where σ is given in (7.67).

We rely much on the proof of Lemma 7.14. It is easy to verify that (7.48) and (7.50) hold here as well; thus, we obtain:

$$\|\widetilde{\eta}''(k+1)\| \leq \left(1 - \frac{\alpha\,\lambda_2(\mathcal{L})}{h_{\max} + \rho}\right)\,\|\widetilde{\eta}''(k)\| + \alpha\,\|x(k+1) - x'(k+1)\|,\ \text{a.s.};$$

after taking expectation:

$$\mathbb{E}\,\left[\|\widetilde{\eta}''(k+1)\|\right] \leq \left(1 - \frac{\alpha\,\lambda_2(\mathcal{L})}{h_{\max} + \rho}\right)\,\mathbb{E}\,\left[\|\widetilde{\eta}''(k)\|\right] + \alpha\,\mathbb{E}\,\left[\|x(k+1) - x'(k+1)\|\right].\tag{7.93}$$

Further, there holds that: $\|\widetilde{\eta}(k)\| \leq \|\widetilde{\eta}''(k)\|$, a.s., and so $\mathbb{E}\,[\|\widetilde{\eta}(k)\|] \leq \mathbb{E}\,[\|\widetilde{\eta}''(k)\|]$. Using the latter and (7.93), and applying Lemma 7.12, we obtain the desired result. Proof of Theorem 7.10 mimics the proof of Theorem 7.3, and is hence omitted.

We now consider the randomized algorithm variant with gradient updates (Algorithm 9 where step (7.61) is replaced with (7.64).) We start with the following Lemma.

Lemma 7.15 (Primal inexactness: Randomized gradient updates) Consider Algorithm 9 where step (7.61)

is replaced with (7.64), and let Assumptions 7.1 and 7.2 hold. Then, for all $k = 0, 1, ...$:

$$\mathbb{E}\left[\left\|x(k+1) - x'(k+1)\right\|\right] \leq e^{-\sigma'\tau}\,\mathbb{E}\left[\|\widetilde{x}(k)\|\right] + e^{-\sigma'\tau}\,\frac{1}{h_{\min}}\,\mathbb{E}\left[\|\widetilde{\eta}(k)\|\right],$$

where σ' is given in (7.74).

Proof of Lemma 7.15 is very similar to the proof of Lemma 7.12. One can easily verify that, with the randomized algorithm and gradient updates, (7.84)–(7.85) hold here with $\frac{\rho^2}{(\rho+h_{\min})^2}$ replaced with $(1 - \beta\, h_{\min})^2$. The proof then proceeds analogously to the proof of Lemma 7.12. Further, just by replacing η with η', we obtain the Lemmas equivalent to Lemmas 7.13 and 7.14, which ultimately confirms the validity of Theorem 7.11.

7.6 Simulation studies

This Section provides a simulation example with the l_2-regularized logistic losses. We summarize our findings from the example. First, simulations corroborate a globally linear convergence of the proposed methods – both deterministic and randomized distributed AL methods. Further, it is usually advantageous to take a small number of inner iterations τ. We also compare: 1) the deterministic AL method with gradient type updates and: 2) the D–NG method in [9], as both methods have computationally inexpensive iterations. Simulations indicate that the AL method is better for smaller (better) condition numbers, while D–NG is better for larger (poorer) condition numbers. (The D–NG method is less sensitive to the condition number γ.) Finally, we compare the D–NG method with the deterministic AL method with NJ updates, which is similar to the D–Lasso method proposed in [8] and show that the two methods trade-off communication and computational costs, irrespective of the condition number. (Al with NJ has a lower communication cost and a larger computational cost.)

Optimization problem. We consider distributed learning via the l_2-regularized logistic loss; see, e.g., [78] for further details. Nodes minimize the logistic loss:

$$\sum_{i=1}^{N} f_i(x) = \sum_{i=1}^{N} \left(\log\left(1 + e^{-b_i(a_i^\top x_1 + x_0)}\right) + \frac{\mathcal{P}\,\|x\|^2}{2\,N} \right),$$

where $\mathcal{P} > 0$ is the regularization parameter, $x = (x_1^\top, x_0)^\top \in \mathbb{R}^{15}$, $a_i \in \mathbb{R}^{14}$ is the node i's feature vector, and $b_i \in \{-1, +1\}$ is its class label. The hessian $\nabla^2 f_i(x) = \frac{\mathcal{P}}{N}\, I + \frac{e^{-c_i^\top x}}{(1+e^{-c_i^\top x})^2} c_i c_i^\top$, where $c_i = (b_i a_i^\top, b_i)^\top \in \mathbb{R}^{15}$. We take node i's constants $h_{\min,i}$ and $h_{\max,i}$ as: $h_{\min,i} = \frac{\mathcal{P}}{N}$ and $h_{\max,i} = \frac{\mathcal{P}}{N} +$

$\frac{1}{4}\|c_i\, c_i^\top\|$. (Note that $\frac{e^{-c_i^\top y}}{(1+e^{-c_i^\top y})^2} \leq 1/4$ for all y.) Further, we let $h_{\min} = \min_{i=1,\ldots,N} h_{\min,i}$ and $h_{\max} = \max_{i=1,\ldots,N} h_{\max,i}$. For the specific problem instance here, the condition number $\gamma = h_{\max}/h_{\min} = 49.55$.

Data. We generate the a_i's independently over i; each entry is drawn from the standard normal distribution. We generate the "true" vector $x^\star = (x_1^{\star\top}, x_0^\star)^\top$ by drawing its entries independently from the standard normal distribution. The class labels are generated as $b_i = \text{sign}\left({x^\star}_1^\top a_i + {x^\star}_0 + \epsilon_i\right)$, where the ϵ_i's are drawn independently from a normal distribution with zero mean and standard deviation 0.001.

Network. The network is a geometric network: nodes are placed uniformly randomly on a unit square and the nodes whose distance is less than a radius are connected by an edge. There are $N = 12$ nodes and 28 links.

Algorithm parameters, metrics, and implementation. We set the weight matrix $W = \frac{1.1}{2} I + \frac{0.9}{2} W_m$, where W_m is the Metropolis weight matrix. (Note that $W \succ 0$.) Further, we set $\alpha = \rho = h_{\min}$, with all algorithm variants, and $\beta = \frac{1}{\rho + h_{\max}} = \frac{1}{(\gamma+1)h_{\min}}$, with the methods that use the gradient primal updates. We set the number of inner iterations τ as follows. For the deterministic variant and NJ updates, we set τ as in the right hand side (rhs) of (7.15); with deterministic+gradient – as in the rhs of (7.21); with randomized+NGS – as in the rhs of (7.70); and with randomized+gradient – as in the rhs of (7.77). We also simulate the methods with $\tau = 1$ (although our theory does not guarantee linear convergence in such case.) We initialize the primal and dual variables with all methods to equal zero. We consider the relative error in the cost function, averaged across nodes, i.e., we estimate $\frac{1}{N}\sum_{i=1}^N \frac{f(x_i) - f^\star}{f(0) - f^\star}$. We compare the methods in terms of: 1) the total number of transmissions (across all nodes), and 2) the total computational time. We implement the methods via a serial implementation – one processor works the jobs of all nodes. We count the CPU time for the overall jobs across all nodes. With the methods that use the NGS and NJ updates in (7.7), we solve the local problems via the fast Nesterov gradient method for strongly convex functions. At the inner iteration s and outer iteration k, to solve (7.7), we initialize the Nesterov gradient method by $x_i(k,s)$. We stop the algorithm after:

$$\left\lceil \left| \frac{\log\left(\frac{2\epsilon}{(R')^2 L'}\right)}{\log(1-\sqrt{\gamma'})} \right| \right\rceil$$

iterations, with $\epsilon = 10^{-5}$.[5] This guarantees that the optimality gap in the cost function upon termination is below $\epsilon = 10^{-5}$. Here, L' is a Lipschitz constant for the cost function in (7.7), that (at node i) we take as $h_{\max,i} + \rho + \frac{\mathcal{R}}{N}$. Further, $\gamma' = L'/\nu'$ is the cost condition number, where $\nu' = \frac{\mathcal{R}}{N} + \rho$ is the Hessian lower bound. Finally, R' is an estimate of the distance to solution, which we take as: $R' = \frac{1}{\rho + \mathcal{R}/N}\|\nabla f_i(x_i(k,s)) + (\mathcal{R}/N + \rho)x_i(k,s) + (\eta_i(k) - \rho \overline{x}_i(k,s))\|$. All our Figures below are in a semi-

[5] We implicitly assume that the physical time allocated for each inner iteration s suffices to perform optimization (7.7).

log scale.

Figure 7.1 (top left) plots the relative error in the cost function for the deterministic variants versus the number of communications, while Figure 7.1 (top right) plots the same quantity versus the CPU time (This is the cumulative CPU time across all nodes.) We simulate the NJ method with τ in (7.15) and $\tau = 1$, and the gradient method with τ in (7.21) and $\tau = 1$. The Figures indicate the linear convergence of the proposed methods. We report that the gradient method with τ in (7.21) also shows a linear convergence in the number of communications, but it converges slowly due to the large value of τ. We can see that the NJ variant is better in terms of the communication cost but is worse in terms of the computational cost. Figures 7.1

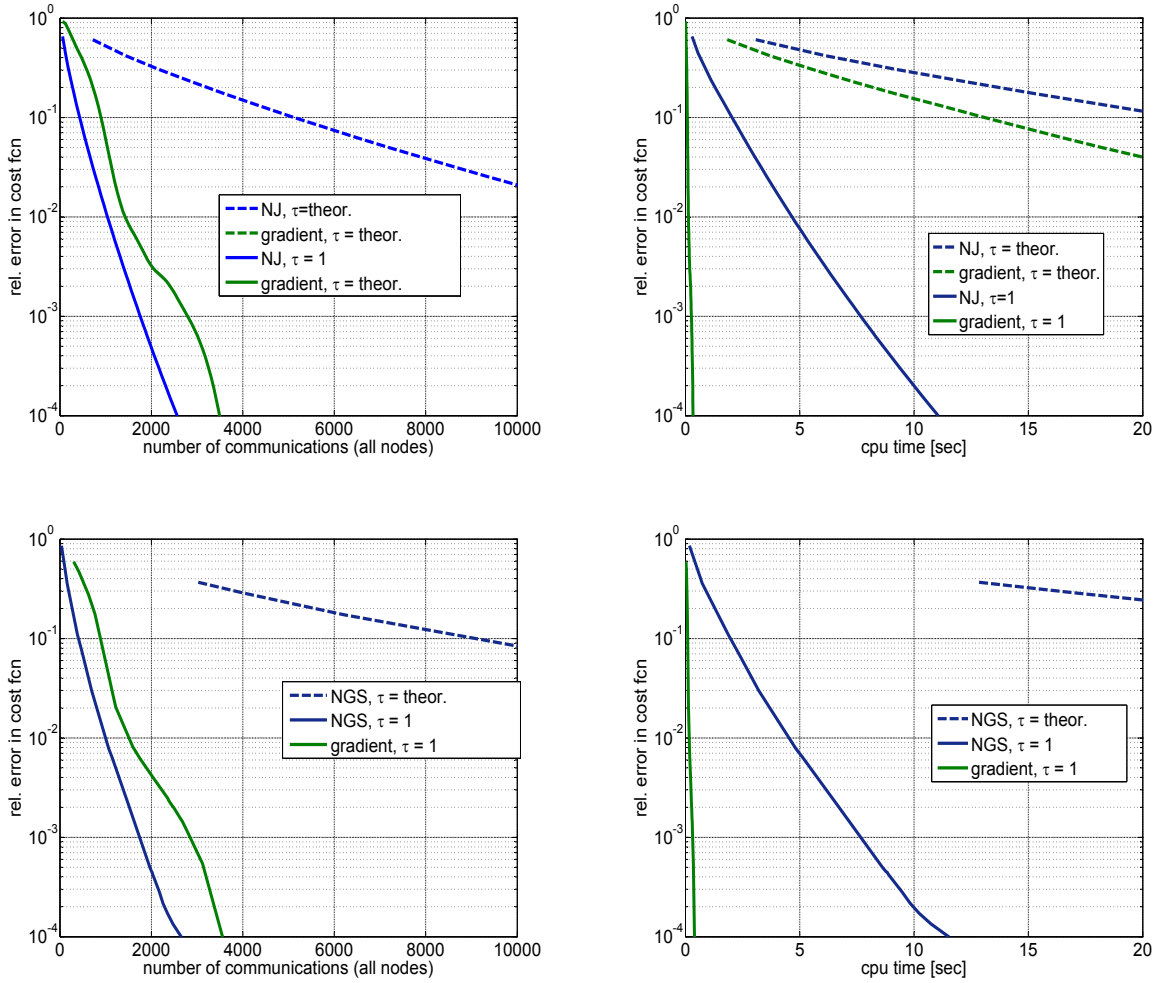

Figure 7.1: Average relative error in the cost function $\frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)-f^\star}{f(0)-f^\star}$ for the proposed deterministic methods (top) and randomized methods (bottom). The two left Figures show the communication cost (total number of communications across all nodes); the two right figures show the computational cost (totol CPU time across all nodes.)

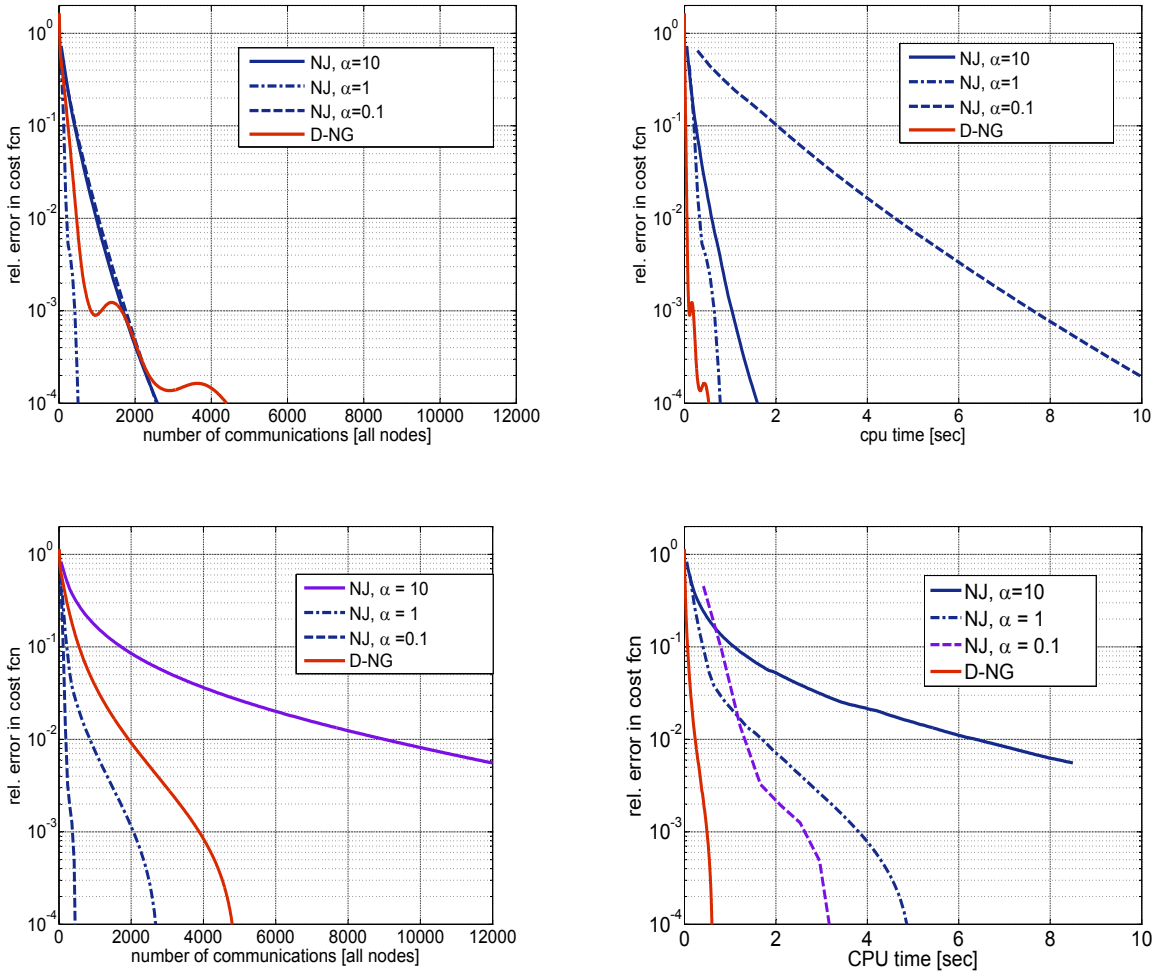

Figure 7.2: Average relative error in the cost function $\frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)-f^\star}{f(0)-f^\star}$ for the deterministic AL with NJ updates method and the D–NG method. Top Figures show the scenario of a smaller condition number $\gamma \approx 49.55$, while bottom Figures show the scenario of a larger condition number $\gamma \approx 4856$. The two left Figures show the communication cost (total number of communications across all nodes); the two right figures show the computational cost (total CPU time across all nodes.)

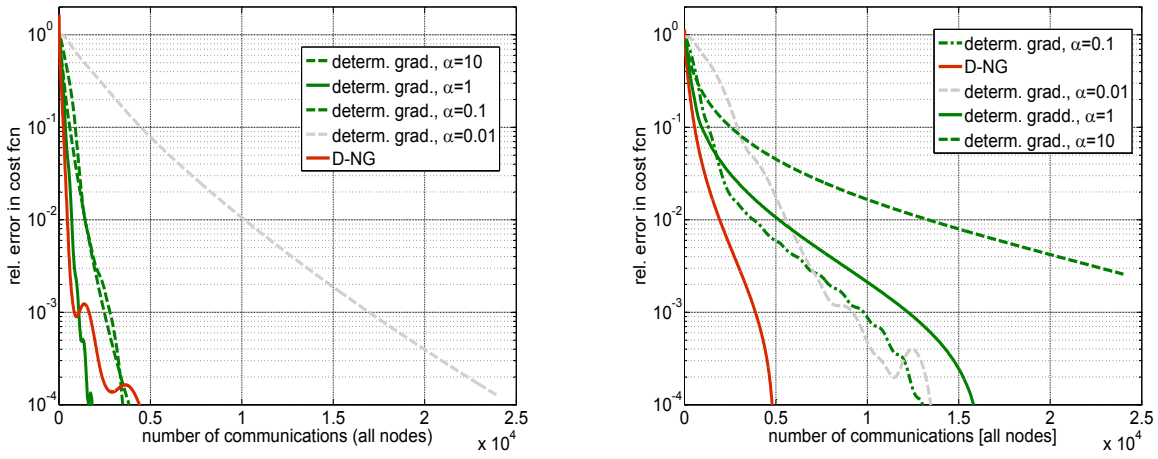

Figure 7.3: Average relative error in the cost function $\frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)-f^\star}{f(0)-f^\star}$ for the deterministic AL method with gradient updates and the D–NG method versus the total number of communications (The methods have a smilar computational cost per communication.) The left Figure shows the scenario of a smaller condition number $\gamma \approx 49.55$, while the right Figure shows the scenario of a larger condition number $\gamma \approx 4856$.

(bottom left and right) make the same plots for the randomized NGS and gradient methods. We observe a similar behavior to that of the deterministic variants. Again, the theoretical value for τ of the randomized gradient method in (7.77) is very large, and, consequently, the algorithm shows slow convergence for the latter choice of τ.

Comparison of deterministic AL with NJ updates and D–NG. In the next experiment, we compare the D–NG method in [9] with the deterministic variant and NJ updates; We consider two scenarios: 1) smaller (better) condition number $\gamma = \frac{h_{\max}}{h_{\min}} = 49.55$; and 2) larger (worse) condition number $\gamma \approx 4856$. The data and network are the same as in the previous example. With the second scenario, we increase the condition number by taking a smaller value of the regularization parameter $\mathcal{P}$. With the AL NJ method, we take $\alpha = \rho \in \{0.01, 0.1, 1, 10\}$, as the optimal choice of α is not known a priori. Figure 7.2 (top left) plots the relative error in the cost function versus the total number of communications, while Figure 7.2 (top right) plots the relative error versus the total CPU time. First, observe that the D–NG method converges sub-linearly in the number of communications. Second, we can see that, in this implementation example, the D–NG has a lower computational cost, while the AL with NJ has a lower communication cost. Further, we can see that D–NG is not very sensitive to the condition number, neither in terms of communication nor in terms of computational costs. Regarding the AL with NJ, it is not very sensitive in terms of the communication cost, but it is sensitive in terms of the computational cost. The reason is that, for a large (poor) condition number γ, the condition number γ' to solve local nodes' problems (7.7) is also poor, and

thus the computational cost increases when γ increases.

Comparison of deterministic AL with gradient updates and D–NG. Our final simulation compares the D–NG method and the deterministic AL with gradient updates. We use the same two scenarios from the previous simulation (with a small and a large condition number γ.) The two methods have a similar (small) computational cost per one communication of each node, and so we only focus on the comparison in terms of the communication cost. With the AL method, we set $\beta = 1/(\rho + h_{\max})$, and we vary $\alpha = \rho \in \{0.01, 0.1, 1, 10\}$. With the AL method, we set the number of inner iterations $\tau = 1$. Figure 7.3 left plots the relative error in the cost function versus the total number of communications for the smaller condition number $\gamma = 49.55$, while Figure 7.3 right repeats the plot for the larger (poorer) condition number $\gamma = 4856$. We can se that the D–NG method is much less sensitive to the condition number γ. The AL method with gradient updates is very sensitive to γ, differently from the AL with NJ updates. We can see that, for a small condition number, the AL gradient method converges faster than D–NG; for a large condition number, D–NG is faster, at least for the accuracies up to 10^{-5}.

7.7 Conclusion

We considered a wide class of both deterministic and randomized distributed augmented Lagrangian (AL) methods, which mutually differ in the primal variable updates. Specifically, we consider: 1) deterministic AL with nonlinear Jacobi updates; 2) deterministic AL with gradient descent; 3) randomized AL with nonlinear Gauss-Seidel; and 4) randomized AL with gradient descent updates. Assuming twice continuously differentiable costs with bounded Hessian, we establish globally linear (geometric) convergence rates of all methods, and we give explicit dependence of the rates on the underlying network parameters. For example, for the deterministic variant with nonlinear Jacobi updates, we establish rate $\mathcal{R} = 1 - \Omega\left(\frac{\lambda_2}{(\gamma+1)}\right)$ in the number of per-node communications, where γ is the Hessian condition number of the f_i's, and λ_2 is the network spectral gap. Simulation examples demonstrate linear convergence of our methods.

Chapter 8

Conclusion

This thesis develops and analyzes distributed optimization algorithms for networked systems, where N nodes in a network minimize the sum $\sum_{i=1}^{N} f_i(x)$ of their locally known, convex costs, subject to a global variable of common interest. This problem formulation encompasses very relevant applications in networked systems, including distributed estimation and source localization in sensor networks, and distributed machine learning.

The main goals of the thesis are the following: 1) to develop novel distributed optimization algorithms; and 2) to establish convergence rate analysis of both newly proposed and existing methods.

Current literature offers two types of distributed methods to solve the above distributed optimization problem, namely distributed (consensus-based) gradient methods, and distributed augmented Lagrangian dual methods. This thesis contributes to both types of methods, mainly in the following.

Distributed gradient-type methods:

- We develop novel distributed gradient methods that converge significantly faster than existing distributed gradient methods;

- We establish global convergence rates of our methods, in terms of the cost function parameters (e.g., Lipschitz constant of the gradient) and the underlying network parameters; remarkably, acceleration techniques guarantee convergence rates (in expectation) on random networks;

- We show that existing distributed gradient methods cannot achieve the rates of our methods under equal network and cost functions conditions.

Distributed augmented Lagrangian (AL) methods:

- We develop novel distributed AL algorithms that operate with asynchronous inter-node communication;

- We establish globally linear convergence rates of a wide class of distributed AL methods under convex twice continuously differentiable costs with bounded Hessian, in terms of the overall per-node communications at any stage of the algorithm.

We recapitulate the main contributions of the thesis on distributed gradient methods, distributed AL methods, and distributed consensus and averaging.

Chapters 2–4: Distributed Nesterov-like gradient methods

We propose distributed Nesterov-like gradient methods, and we establish their convergence rate guarantees for both static and random networks, thus handling random packet dropouts with wireless sensor networks and asynchronous communication protocols. In Chapters 2 and 3, we achieve this on the class $\mathcal{F}$ of convex, differentiable costs f_i's that have Lipschitz continuous and bounded gradients. Chapter 4 establishes convergence rates for alternative function classes, hence further broadening the applications scope.

In Chapter 2, we consider static networks and the class $\mathcal{F}$ and propose two distributed Nesterov-like methods. Our first method, termed Distributed Nesterov Gradient method (D–NG), achieves at any node i the following convergence rates in the optimality gap at the cost function $\frac{1}{N}\,(\,f(x_i) - f^\star\,)$:

$$O\left(\frac{1}{(1-\mu)^{1+\xi}}\frac{\log k}{k}\right) \text{ and } O\left(\frac{1}{(1-\mu)^{1+\xi}}\frac{\log \mathcal{K}}{\mathcal{K}}\right),$$

in the number of per-node communications $\mathcal{K}$ and per-node gradient evaluations k. Here, $(1-\mu) \in (0,1]$ is the network's spectral gap, and $\xi > 0$ is arbitrarily small.

Our second method, termed Distributed Nesterov gradient with Consensus iterations (D–NC), achieves convergence rates:

$$O\left(\frac{1}{(1-\mu)^2}\frac{1}{\mathcal{K}^{2-\xi}}\right) \text{ and } O\left(\frac{1}{k^2}\right).$$

Both distributed gradient methods D–NG and D–NC show significant gains over existing, standard distributed gradient methods [2], for which we show they cannot perform better than $\Omega\left(\frac{1}{k^{2/3}}\right)$ and $\Omega\left(\frac{1}{\mathcal{K}^{2/3}}\right)$.

In Chapter 3, we modify our D–NG and D–NC methods to handle *random networks*, modeled by a sequence of independent, identically distributed matrices $W(k)$, drawn from the set of symmetric, stochastic matrices with positive diagonals.

We refer to our modified methods as mD–NG and mD–NC, respectively. With both methods, we establish convergence rates in terms of the *expected* normalized optimality gap $\frac{1}{N}\left(\mathbb{E}\left[f(x_i)\right]-f^\star\right)$, at arbitrary node i, as a function of k, $\mathcal{K}$, the number of nodes N, and the quantity $1-\overline{\mu}$ – a generalization of the spectral gap $1-\mu$ for random networks.

The mD–NG algorithm achieves rates

$$O\left(\frac{N}{(1-\overline{\mu})^{4/3}}\frac{\log k}{k}\right) \text{ and } O\left(\frac{N}{(1-\overline{\mu})^{4/3}}\frac{\log \mathcal{K}}{\mathcal{K}}\right),$$

while mD–NC achieves:

$$O\left(\frac{N^\xi}{(1-\overline{\mu})^2}\frac{1}{\mathcal{K}^{2-\xi}}\right) \text{ and } O\left(\frac{1}{k^2}\right).$$

Hence, we show that the acceleration ideas of Nesterov apply also to random networks and allow for much faster algorithms than offered by the existing literature.

In Chapter 4, we establish convergence and convergence rate guarantees for our distributed gradient methods under problem classes different than $\mathcal{F}$. We do not explicitly require that the gradients be bounded, and we allow for constrained optimization, where each node i has the same closed, convex constraint set $\mathcal{X}$. For our proposed methods, naturally adapted to constrained optimization, we establish the same rates as under class $\mathcal{F}$, in terms of per-node communications $\mathcal{K}$ and per-node gradient evaluations k.

Chapter 5: Weight Optimization for Consensus in Random Networks

We address the problem of the *optimal weight design* for consensus averaging algorithms, allowing for random networks with spatially correlated link failures. Our weight design applies both to 1) consensus and distributed averaging; and 2) distributed optimization methods, where the convergence constant depends on the underlying consensus dynamics.

We address the weight design for both symmetric and asymmetric random links. With symmetric random links, we use as the optimization criterion the mean squared consensus convergence rate that equals $\overline{\mu}^2$. We express the rate as a function of the link occurrence probabilities, their correlations, and the weights. We prove that $\overline{\mu}^2$ is a convex, nonsmooth function of the weights, enabling global optimization of the weights. We provide insights how the optimal weights depend on the number of nodes, the link occurrence probabilities, and their correlations. We extend our results to asymmetric random links, adopting as an optimization criterion the mean squared deviation (from the current average state) rate, and show that this metric is a convex function of the weights. Simulation examples demonstrate the gains with our weight design compared with existing weight assignment choices, both in distributed averaging and in distributed optimization.

Chapters 6 and 7: Distributed augmented Lagrangian (AL) methods

We propose novel distributed AL methods that utilize asynchronous inter-node communication. For well-structured convex costs f_i's, we establish globally linear convergence rates of wide class of distributed AL methods. We now recapitulate our contributions in more detail.

In Chapter 6, we propose a randomized distributed AL method, termed Augmented Lagrangian algorithm with Gossip communication (AL–G). The algorithm handles very general, nondifferentiable costs f_i's, private constraint sets, and utilizes asynchronous, unidirectional, gossip communication. With respect to the literature, our AL–G method uses a novel, asynchronous algorithm to update the primal variables. When translated into optimization terminology, this is the nonlinear Gauss-Seidel method with the randomized order of updates. We prove convergence of this inner primal algorithm, when the number of inner iterations goes to infinity. This establishes convergence of the nonlinear Gauss-Seidel method with *random* order of minimizations, while existing literature previously showed convergence only under the *cyclic* or the *essentially cyclic* rules, [14, 15]. We illustrate the performance of our AL–G method with relevant applications in l_1–regularized logistic regression for classification and cooperative spectrum sensing for cognitive radios.

In Chapter 7, with respect to Chapter 6, we assume a restricted class of functions of convex, twice differentiable f_i's with a bounded Hessian, and unconstrained problems. We establish globally linear convergence rates for a wide class of distributed AL methods, in the overall number of per-node communications $\mathcal{K}$ at any algorithm stage. Furthermore, we give explicit dependence of the convergence rate on the network spectral gap $1-\mu$.

Specifically, we analyze a wide class of both deterministic and randomized methods that update their dual variables at slow time scale and their primal variables iteratively, at a fast time scale. With the deterministic variants, primal variables are updated via either: 1) the nonlinear Jacobi (NJ) method, or 2) the gradient descent. With the randomized methods, primal variables are updated via either: 1) a randomized nonlinear Gauss-Seidel (NGS) method; or a randomized coordinate gradient descent. Hence, we consider the total of four algorithm variants: 1) deterministic NJ; 2) deterministic gradient; 3) randomized NGS; and 4) randomized gradient. With all four variants, we establish globally linear convergence rates in the total number of elapsed per-node communications $\mathcal{K}(k)$ after k outer iterations. The distance to the solution $x^\star$ of (1.1), at any node i and any outer iteration k decays as:

$$\|x_i(k) - x^\star\| = O\left(\mathcal{R}^{\mathcal{K}(k)}\right),$$

where $\mathcal{R} \in [0,1)$ is the communication rate. We explicitly express the rate $\mathcal{R}$ in terms of the f_i's condition

number γ and the network spectral gap. For example, for the deterministic NJ method, the rate is:

$$\mathcal{R} = 1 - \Omega\left(\frac{1-\mu}{\gamma}\right)$$

which is $1 - \Omega(1/N^2)$ (poor) for chain networks and is bounded away from one (good) for expander networks.

The thesis develops new technical tools that are of general interest in classical and distributed optimization, linear random time varying systems, and random consensus dynamics.

Future work

Directions for future work include the following. 1) Explore convergence rates of distributed Nesterov-like methods with composite, nondifferentiable costs; 2) Explore convergence rates of distributed augmented Lagrangian methods for more general costs (than the costs with bounded Hessian). 3) Consider distributed optimization formulations with costs different than the sum of nodes' local costs.

Portions of this thesis have been published in journal papers [46, 38], submitted to journals [9, 47], and are to be submitted to a journal [48]; and published in conference proceedings [49, 50].

During the course of this thesis, we also published journal [3, 51, 52] and conference [53, 54, 55, 56, 57, 58] papers.

Appendix A

Technical Proofs for Chapter 2

A.1 Proof of Lemma 2.5

Step 1. We first prove the following auxiliary equality:

$$\gamma_{k-1}\overline{v}(k) = \overline{x}(k) - (1-\gamma_{k-1})\overline{x}(k-1). \tag{A.1}$$

Using the definition of $\overline{v}(k)$ in Lemma 2.5, $\gamma_k = 2/(k+2)$, $\beta_{k-1} = (k-1)/(k+2)$, and (4.51):

$$\overline{v}(k) = \frac{k+2}{2}\left(\overline{x}(k) + \frac{k-1}{k+2}\overline{x}(k) - \frac{k-1}{k+2}\overline{x}(k-1) - \frac{k}{k+2}\right) = \frac{k+1}{2}\overline{x}(k) - \frac{k-1}{2}\overline{x}(k-1).$$

Multiplying the right hand side of the last equality by $\gamma_{k-1} = 2/(k+1)$, the result follows.

Step 2. We prove the following relation:

$$f(\overline{x}(k)) \leq f(z) + L_{k-1}(\overline{x}(k) - \overline{y}(k-1))^\top (z - \overline{x}(k)) + \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 + \delta_{k-1},\ \forall z \in \mathbb{R}^d. \tag{A.2}$$

Using the inexact oracle property (4.49):

$$f(\overline{x}(k)) \leq \widehat{f}_{k-1} + \widehat{g}_{k-1}^\top (\overline{x}(k) - \overline{y}(k-1)) + \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 + \delta_{k-1}. \tag{A.3}$$

Further, $\forall z \in \mathbb{R}^d$: $0 = 0^\top (z - \overline{x}(k)) = \left(\overline{x}(k) - \overline{y}(k-1) + \frac{1}{L_{k-1}}\widehat{g}_{k-1}\right)^\top (z - \overline{x}(k))$, and so:

$$\widehat{g}_{k-1}^\top (z - \overline{x}(k)) + L_{k-1}(\overline{x}(k) - \overline{y}(k-1))^\top (z - \overline{x}(k)) = 0. \tag{A.4}$$

From property (4.49): $f(z) \geq \widehat{f}_{k-1} + \widehat{g}_{k-1}^{\top}(z - \overline{y}(k-1))$, and so, using the last equation and adding (A.3) and (A.4), the claim (A.2) follows.

Step 3. We finally prove (4.53). We start by using relation (A.2). Namely: 1) setting $z = \overline{x}(k-1)$ in (A.2) and multiplying inequality (A.2) by $1 - \gamma_{k-1}$; 2) setting $z = x^{\bullet}$ in (A.2) and multiplying inequality (A.2) by γ_{k-1}; and 3) adding the corresponding two inequalities:

$$\begin{aligned}
& \gamma_{k-1}\left\{f(\overline{x}(k)) - f(x^{\bullet})\right\} + (1-\gamma_{k-1})\left\{f(\overline{x}(k)) - f(\overline{x}(k-1))\right\} \\
= \;& \left\{f(\overline{x}(k)) - f(x^{\bullet})\right\} - (1-\gamma_{k-1})\left\{f(\overline{x}(k-1)) - f(x^{\bullet})\right\} \\
\leq \;& \gamma_{k-1}L_{k-1}\,(\overline{x}(k) - \overline{y}(k-1))^{\top}(x^{\bullet} - \overline{x}(k)) + (1-\gamma_{k-1})L_{k-1}\,(\overline{x}(k) - \overline{y}(k-1))^{\top}(\overline{x}(k-1) - \overline{x}(k)) \\
+ \;& \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 + \delta_{k-1} \\
= \;& L_{k-1}(\overline{x}(k) - \overline{y}(k-1))^{\top}(\gamma_{k-1}x^{\bullet} + (1-\gamma_{k-1})\overline{x}(k-1) - \overline{x}(k)) + \frac{L_{k-1}}{2}\|\overline{x}(k) - \overline{y}(k-1)\|^2 + \delta_{k-1} \\
= \;& \frac{L_{k-1}}{2}(2(\overline{x}(k) - \overline{y}(k-1))^{\top}(\gamma_{k-1}x^{\bullet} + (1-\gamma_{k-1})\overline{x}(k-1) - \overline{x}(k)) + \|\overline{x}(k) - \overline{y}(k-1)\|^2) + \delta_{k-1} \quad \text{(A.5)}
\end{aligned}$$

Denote by:

$$\mathcal{M}_{k-1} = (2(\overline{x}(k) - \overline{y}(k-1))^{\top}(\gamma_{k-1}x^{\bullet} + (1-\gamma_{k-1})\overline{x}(k-1) - \overline{x}(k)) + \|\overline{x}(k) - \overline{y}(k-1)\|^2).$$

Then, inequality (A.5) is written simply as:

$$\left\{f(\overline{x}(k)) - f(x^{\bullet})\right\} - (1-\gamma_{k-1})\left\{f(\overline{x}(k-1)) - f(x^{\bullet})\right\} \leq \frac{L_{k-1}}{2}\mathcal{M}_{k-1} + \delta_{k-1}. \quad \text{(A.6)}$$

Now, we simplify the expression for $\mathcal{M}_{k-1}$ as follows. Using the identity:

$$\|\overline{x}(k) - \overline{y}(k-1)\|^2 = 2(\overline{x}(k) - \overline{y}(k-1))^{\top}\overline{x}(k) + \|\overline{y}(k-1)\|^2 - \|\overline{x}(k)\|^2,$$

we have:

$$\begin{aligned}
\mathcal{M}_{k-1} &= 2(\overline{x}(k) - \overline{y}(k-1))^{\top}(\gamma_{k-1}x^{\bullet} + (1-\gamma_{k-1})\overline{x}(k-1)) - \|\overline{x}(k)\|^2 + \|\overline{y}(k-1)\|^2 \\
&= \|\overline{y}(k-1) - ((1-\gamma_{k-1})\overline{x}(k-1) + \gamma_{k-1}x^{\bullet})\|^2 - \|\overline{x}(k) - ((1-\gamma_{k-1})\overline{x}(k-1) + \gamma_{k-1}x^{\bullet})\|^2 \\
&= \gamma_{k-1}^2\|\overline{v}(k-1) - x^{\bullet}\|^2 - \gamma_{k-1}^2\|\overline{v}(k) - x^{\bullet}\|^2, \quad \text{(A.7)}
\end{aligned}$$

where the last equality follows by the definition of $\overline{v}(k-1)$ in Lemma 2.5 and by the identity (A.1). Now,

combining (A.6) and (A.7):

$$\begin{aligned}(f(\overline{x}(k)) - f(x^\bullet)) &- (1-\gamma_{k-1})(f(\overline{x}(k-1)) - f(x^\bullet)) \\ &\leq \frac{L_{k-1}\gamma_{k-1}^2}{2}\left(\|\overline{v}(k-1) - x^\bullet\|^2 - \|\overline{v}(k) - x^\bullet\|^2\right) + \delta_{k-1}.\end{aligned}$$

Finally, multiplying the last equation by $\frac{4}{\gamma_{k-1}^2}$, and using $\gamma_{k-1} = 2/(k+1)$, we get the result.

A.2 Proof of Theorem 2.8 (b)

Suppose $c > \frac{1}{2L}$ and denote by $k' = 2cL$. We show that Theorem 5 (b) holds with:

$$\begin{aligned}\mathcal{C}' &= k'L\left(\left(\frac{3^{k'}-1}{3-1}\right)^2 4c^2G^2 + R^2\right) + \frac{2}{c}\left(2(2k'+1)^2\left(\frac{3^{k'}-1}{3-1}\right)^2 4c^2G^2 + 2R^2\right) \\ &+ 16c^2LC_{\text{cons}}^2G^2 + c\,C_{\text{cons}}G^2. \end{aligned} \tag{A.8}$$

Progress equation (2.24) still holds if $L'_k = \frac{N(k+1)}{c} \geq 2NL$, i.e., if $k \geq k' := 2cL$. Telescoping (2.24) backwards from $k > k'$ until k':

$$\begin{aligned}&\frac{(k+1)^2-1}{k+1}\left(f(\overline{x}(k)) - f^\star\right) \\ &\leq \frac{(k')^2-1}{k'}\left(f(\overline{x}(k'-1)) - f^\star\right) + \frac{2N}{c}\|\overline{v}(k'-1) - x^\star\|^2 + L\sum_{t=k'}^{k}\|\widetilde{y}(t-1)\|^2\frac{(t+1)^2}{t} \\ &\leq k'\left(f(\overline{x}(k'-1)) - f^\star\right) + \frac{2N}{c}(2\|\overline{v}(k'-1)\|^2 + 2\|x^\star\|^2) + L\sum_{t=1}^{k}\|\widetilde{y}(t-1)\|^2\frac{(t+1)^2}{t}. \end{aligned} \tag{A.9}$$

Theorem 2.7 holds unchanged if $c > 1/(2L)$, and so:

$$L\sum_{t=1}^{k}\|\widetilde{y}(t-1)\|^2\frac{(t+1)^2}{t} \leq \frac{16N\,c^2}{k}LC_{\text{cons}}^2G^2\left(\sum_{t=2}^{k}\frac{(t+2)^2}{t(t-1)^2}\right). \tag{A.10}$$

Upper bound $\|\overline{v}(k'-1)\|$, where we recall $\overline{v}(k) = \frac{1}{\gamma_k}\overline{y}(k) - \frac{1-\gamma_k}{\gamma_k}\overline{x}(k)$, and $\gamma_k = \frac{2}{k+2}$:

$$\|\overline{v}(k-1)\| \leq \frac{k+1}{2}\|\overline{y}(k-1)\| + k\|\overline{x}(k-1)\| \leq (2k+1)M_{k-1}, \tag{A.11}$$

where $M_{k-1} := \max\left\{\|\overline{y}(k-1)\|,\, \|\overline{x}(k-1)\|\right\}$. By (31), $\alpha_{k-1} = c/k$, $\beta_{k-1} \leq 1$, $\|\sum_{i=1}^{N}\nabla f_i(y_i(k-$

$1))\| \leq NG$, $\|W\| = 1$:

$$\begin{aligned}\|\overline{x}(k)\| &\leq \|\overline{y}(k-1)\| + \frac{cG}{k} \leq M_{k-1} + \frac{cG}{k} \\ \|\overline{y}(k)\| &\leq 2\|\overline{x}(k)\| + \|\overline{x}(k-1)\| \leq 2\|\overline{y}(k-1)\| + \|\overline{x}(k-1)\| + \frac{2cG}{k} \leq 3M_{k-1} + \frac{2cG}{k},\end{aligned}$$

and so: $M_k \leq 3M_{k-1} + \frac{2cG}{k},\ k = 1, 2, ..., \quad M_0 = 0$. By unwinding the above recursion from $k = k' - 1$ to $k = 1$, we get:

$$M_{k'-1} \leq \left(\frac{3^{k'} - 1}{3 - 1}\right) 2cG. \tag{A.12}$$

Further, combining the last equation with (A.11), and squaring the resulting inequality:

$$\|\overline{v}(k'-1)\|^2 \leq (2k'+1)^2 \left(\frac{3^{k'} - 1}{3 - 1}\right)^2 4c^2G^2. \tag{A.13}$$

Using the Lipschitz continuity of f (with constant LN), and using $\nabla f(x^\star) = 0$:

$$\begin{aligned}f(\overline{x}(k'-1)) - f^\star &\leq \nabla f(x^\star)^\top(\overline{x}(k'-1) - x^\star) + \frac{LN}{2}\|\overline{x}(k'-1) - x^\star\|^2 \\ &\leq (LN)\left(\|\overline{x}(k'-1)\|^2 + \|x^\star\|^2\right) \leq (LN)\left(\left(\frac{3^{k'} - 1}{3 - 1}\right)^2 4c^2G^2 + \|x^\star\|^2\right),\end{aligned}$$

where we used (A.12) to upper bound $\|\overline{x}(k'-1)\|$. Finally, combine the last equation with (A.9), (A.10), (A.13), use $\sum_{t=1}^{k} \frac{(t+1)^2}{t^3} \geq 1$, $\forall k$, $\|x^\star\| \leq R$ (as $x(0) = y(0) = 0$), repeat the same argument as in (2.26). We obtain:

$$\frac{1}{N}\left(f(x_i(k)) - f^\star\right) \leq \mathcal{C}'\left(\frac{1}{k}\sum_{t=2}^{k} \frac{(t+2)^2}{t(t-1)^2}\right), \quad k > k',$$

where $\mathcal{C}'$ is given in (A.8).

A.3 Auxiliary steps for the proof of the lower bound (2.34)

Proof of Step 1: Properties of the f_i^θ's. We now show that the f_i^θ's are convex, have Lipschitz continuous gradient with constant $L = \sqrt{2}$, and bounded gradients $\|\nabla f_i^\theta(x)\| \leq 10$, for all x, $i = 1, 2$. Thus, the f_i^θ's in (2.35) belong to the class $\mathcal{F} = \mathcal{F}(L = \sqrt{2}, G = 10)$, for any $\theta \in [0, 1]$.

To show that the function $x \mapsto f_1^\theta(x)$ is convex, note that it can be represented as the following concatenation: $x \mapsto y = (\sqrt{\theta}(x^{(1)} - 1),\ (x^{(2)} - 1))^\top \mapsto z = \|y\| \mapsto w = f_h(z) = f_1^\theta(x)$, where $f_h : \mathbb{R}_+ \to \mathbb{R}$

is the Huber loss: $f_h(z) = \frac{1}{2}z^2$, if $\|z\| \leq \overline{\chi}$, and $f_h(z) = \overline{\chi}(\|z\| - \overline{\chi}/2)$, else. Hence, $x \mapsto f_1^\theta(x)$ is a concatenation of an affine function, a convex function, and a convex non-decreasing function, and hence it is convex. Analogously, we can show that $x \mapsto f_2^\theta(x)$ is convex.

We show the Lipschitz continuity and the boundedness of the gradient of f_1^θ:

$$\nabla f_1^\theta(x) = \left(\frac{\partial f_1^\theta}{\partial x^{(1)}}(x),\ \frac{\partial f_1^\theta}{\partial x^{(2)}}(x)\right)^\top = \begin{cases} (\theta(x^{(1)}-1),\ (x^{(2)}-1))^\top & \text{if } x \in \mathcal{R}_1 \\ \frac{\overline{\chi}}{\left[\theta(x^{(1)}-1)^2+(x^{(2)}-1)^2\right]^{1/2}}(\theta(x^{(1)}-1),\ (x^{(2)}-1))^\top & \text{else.} \end{cases} \tag{A.14}$$

The first coordinate of the gradient $x \mapsto \frac{\partial f_1^\theta}{\partial x^{(1)}}(x)$ can be expressed as the following concatenation of the functions: $x \mapsto y = (x^{(1)}-1, x^{(2)}-1)^\top \mapsto z = (\sqrt{\theta}y^{(1)}, y^{(2)}) \mapsto w = \text{Proj}_{B_{0,\overline{\chi}}}(z) \mapsto v = \frac{\partial f_1^\theta}{\partial x^{(1)}}(x) = \sqrt{\theta}w^{(1)}$, where $\text{Proj}_{B_{0,\overline{\chi}}}(z)$ is the projection of z on the ball centered at zero with radius $\overline{\chi}$. All the functions ϕ in the concatenation above are Lipschitz continuous with constant one, and so $x \mapsto \frac{\partial f_1^\theta}{\partial x^{(1)}}$ is also Lipschitz continuous with constant one. (Given the function $\phi_m(\phi_{m-1}(...(\phi_1(x))))$, where the ϕ_i's are Lipschitz continuous of constant one, we have $\|\phi_m(\phi_{m-1}(...(\phi_1(u)))) - \phi_m(\phi_{m-1}(...(\phi_1(v))))\| \leq \|\phi_{m-1}(...(\phi_1(u))) - \phi_{m-1}(...(\phi_1(v)))\| \leq ...\|u-v\|$.) Similarly, we can show that $x \mapsto \frac{\partial f_1^\theta}{\partial x^{(2)}}$ is Lipschitz continuous with constant one. This implies that the gradient $x \mapsto \nabla f_1^\theta(x)$ is Lipschitz continuous with constant $\sqrt{2}$. Also, $\|\frac{\partial f_1^\theta}{\partial x^{(1)}}(x)\| \leq \sqrt{\theta}\overline{\chi} \leq 6$, for all x. (Recall the concatenation representation $x \mapsto y = (x^{(1)}-1, x^{(2)}-1)^\top \mapsto z = (\sqrt{\theta}y^{(1)}, y^{(2)}) \mapsto w = \text{Proj}_{B_{0,\overline{\chi}}}(z) \mapsto v = \frac{\partial f_1^\theta}{\partial x^{(1)}}(x) = \sqrt{\theta}w^{(1)}$; then, for any $x \in \mathbb{R}^2$, $\|\frac{\partial f_1^\theta}{\partial x^{(1)}}(x)\| \leq \sqrt{\theta}\|\text{Proj}_{B_{0,\overline{\chi}}}(z)\|$, for some $z \in \mathbb{R}^2$, and so $\|\frac{\partial f_1^\theta}{\partial x^{(1)}}(x)\| \leq \sqrt{\theta}\overline{\chi}$.) Similarly, $\|\frac{\partial f_1^\theta}{\partial x^{(2)}}(x)\| \leq \overline{\chi} \leq 6$. Thus, for the gradient, we have: $\|\nabla f_1^\theta(x)\| \leq 6\sqrt{2} < 10$, for all x. We can analogously show that $\|\nabla f_2^\theta(x)\| \leq 6\sqrt{2} < 10$, for all x.

Proof of (2.53). We fist prove that, if $\|x^{\text{I}}(k)\| \leq 2\sqrt{2}$, and $\|x^{\text{II}}\| \leq 2\sqrt{2}$, then $x_i(k) \in \mathcal{R}_i$, $i = 1, 2$. Consider node 1's estimate $x_1(k)$. If $\|x^{\text{I}}(k)\| \leq 2\sqrt{2}$, and $\|x^{\text{II}}\| \leq 2\sqrt{2}$, then $\|x_1^{(l)}(k)\| \leq 2\sqrt{2}$, $l = 1, 2$, and:

$$\theta(x_1^{(1)}(k)-1)^2 + (x_1^{(2)}(k)-1)^2 \leq 2(2\sqrt{2}+1)^2 < 2(2\frac{3}{2}+1)^2 < 32 < \overline{\chi}^2 = 36,$$

which means $x_1(k) \in \mathcal{R}_1$. (Analogously, we can show $x_2(k) \in \mathcal{R}_2$.)

We next prove that $\|x^l(k)\| \leq 2\sqrt{2}$, $l = 1, 2$, for all k; we do this by induction. For $k = 0$, $\|x^l(0)\| \leq 2\sqrt{2}$, $l = 1, 2$. Now, suppose that, for some $k \geq 1$, $\|x^l(k-1)\| \leq 2\sqrt{2}$, $l = 1, 2$. Then, the update equations (2.52) to get $x^{\text{I}}(k)$ and $x^{\text{II}}(k)$, using the derivatives of the f_i^θ's in the quadratic region in (A.14) are given by (2.54). From (2.54), the sub-additive and sub-multiplicative properties of norms, and using $\alpha_{k-1} =$

$c/(k^\tau)$:

$$\|x^{\mathrm{I}}(k)\| \leq \left(1 - \frac{c\theta}{k^\tau}\right)\|x^{\mathrm{I}}(k-1)\| + \frac{c\theta}{k^\tau}\sqrt{2} = \|x^{\mathrm{I}}(k-1)\| - \frac{\theta c}{k^\tau}\left(\|x^{\mathrm{I}}(k-1)\| - \sqrt{2}\right).$$

Now, distinguish two cases: 1) $\|x^{\mathrm{I}}(k-1)\| \in [0, \sqrt{2}]$; and 2) $\|x^{\mathrm{I}}(k-1)\| \in (\sqrt{2}, 2\sqrt{2}]$. In case 1: $\|x^{\mathrm{I}}(k)\| \leq \|x^{\mathrm{I}}(k-1)\| + \frac{\sqrt{2}\theta c}{k^\tau} \leq 2\sqrt{2}$, where we used $0 \leq c \leq 1/(2\sqrt{2}) = 1/(2L)$ and $0 \leq \theta \leq 1$. In case 2: $\|x^{\mathrm{I}}(k)\| < \|x^{\mathrm{I}}(k-1)\| \leq 2\sqrt{2}$. Thus, we have shown that $\|x^{\mathrm{I}}(k)\| \leq 2\sqrt{2}$. Similarly, we can show that $\|x^{\mathrm{II}}(k)\| \leq 2\sqrt{2}$. Thus, by induction, $\|x^l(k)\| \leq 2\sqrt{2}$, $l = 1, 2$, for all k, and so $x_i(k) \in \mathcal{R}_i$, $i = 1, 2$, for all k.

Proof of an inequality on $s_k(\tau)$. Consider function $s_k : [0,1] \to \mathbb{R}$, $s_k(\tau) = \sum_{t=0}^{k-1}(t+1)^{-\tau}$. We prove that $s_k(\tau) \leq 3(\log k)k^{1-\tau}$, $\tau \in [0,1]$, $k \geq 3$. The function is convex on $[0,1]$. By convexity, $s_k(1) \geq s_k(\tau) + \nabla s_k(\tau)(1-\tau)$, and so:

$$\begin{aligned} s_k(\tau) &\leq s_k(1) + (\tau-1)\nabla s_k(\tau) = s_k(1) + (\tau-1)\left(-\sum_{t=2}^{k}(\log t)t^{-\tau}\right) \\ &\leq s_k(1) + (1-\tau)(\log k)\left(\sum_{t=2}^{k} t^{-\tau}\right) \\ &\leq s_k(1) + (1-\tau)(\log k)\frac{(k+1)^{1-\tau}}{1-\tau} \leq (3\log k)(k+1)^{1-\tau}, \end{aligned} \tag{A.15}$$

for all $k \geq 3$. In the left ineuality in (A.15), we use $\sum_{t=2}^{k} t^{-\tau} \leq \frac{(k+1)^{1-\tau}}{1-\tau}$, while in the right inequality we use $s_k(1) \leq \log k + 1$ and $\log k \geq 1$ for $k \geq 3$.

Finding the infima over $\tau \in [0, 3/4]$**,** $\tau \in [3/4, 1]$**, and** $\tau \in [1, \infty)$ **in** (2.60). First, upper bound $\inf_{[3/4,1]} e_k(\tau)$. Using $s_k(\tau) \leq 3(\log k)(k+1)^{1-\tau}$, $\forall k \geq 3$, $\forall \tau \in [0,1]$, and (2.59): $e_k(\tau) \geq e'_k(\tau) = \frac{(1-c_{\max})^2}{6(\log k)(k+1)^{1-\tau}} + \frac{c_{\min}^2}{2(\log k)(k+1)^{2\tau}}$, $\forall k \geq 3$, $\forall \tau \in [3/4, 1]$. Thus:

$$\inf_{[3/4,1]} e_k(\tau) = \Omega\left(\frac{1}{(\log k)(k+1)^{1/4}}\right), \tag{A.16}$$

after setting $\tau = 3/4$. Next, consider $\tau \in [0, 3/4]$: upper bound $e_k(\tau)$ using $s_k(\tau) \leq 1 + \frac{(k+1)^\tau}{1-\tau} \leq 1 + 4(k+1)^\tau$, $\tau \in [0, 3/4]$ and (2.59): $e_k(\tau) \geq e''_k(\tau) := \frac{(1-c_{\max})^2}{2+8\,(k+1)^{1-\tau}} + \frac{c_{\min}^2}{2(k+1)^{2\tau}}$, $\forall k \geq 1$, $\forall \tau \in [0, 3/4]$, and thus we obtain:

$$\inf_{[0,3/4]} e_k(\tau) = \Omega(1/k^{2/3}). \tag{A.17}$$

Finally, consider $\tau \in [1, \infty)$; we have $e_k(\tau) \geq \frac{(1-c_{\max})^2}{2s_k(\tau)} \geq \frac{(1-c_{\max})^2}{6\log k}$, $\forall \tau \in [1, \infty)$, where we used

$s_k(\tau) \le 3\log k, \forall k \ge 3, \forall \tau > 1$. Thus, $\inf_{[1,\infty)} e_k(\tau) = \Omega(1/\log k)$.

A.4 Relaxing bounded gradients: Proof of (2.36) for D–NG

Consider the candidate weight choice $W_{12} = W_{21} = 1 - W_{11} = 1 - W_{22} = \frac{1}{2}(1 - 10^{-6})$. The eigenvalues of W are $\lambda_1 = 10^{-6}$ and $\lambda_2 = 1$,and so W obeys Assumption 2.1 (b). For the proof of (2.36), start similarly as with D–NC: 1) find the recursion for $x(k), y(k)$ from (2.9)–(2.10) and taking the derivatives of the f_i's in (2.37), with $x(0) = y(0) = (0,0)^\top$; 2) define $z(k) := Q^\top x(k)$, $w(k) := Q^\top y(k)$; and 3) write the recursion for the first coordinate $z^{(1)}(k), w^{(1)}(k)$, using $\alpha_k = c/(k+1)$:

$$z^{(1)}(k) = \left(\lambda_1 - \frac{c}{k}\right) w^{(1)}(k-1) + \frac{\sqrt{2}\, c\theta}{k}, \;\; w^{(1)}(k) = z^{(1)}(k) + \beta_{k-1}(z^{(1)}(k) - z^{(1)}(k-1)), \quad \text{(A.18)}$$

$k = 1, 2, ...$ and $z^{(1)}(0) = w^{(1)}(0) = 0$. The update for $(z^{(1)}(k), z^{(1)}(k-1))^\top$ is:

$$(z^{(1)}(k), z^{(1)}(k-1))^\top = \Sigma_1'(k-1)\,(z^{(1)}(k-1), z^{(1)}(k-2))^\top + \frac{c}{k}\,(\sqrt{2}\theta, 0)^\top, \quad \text{(A.19)}$$

$k = 1, 2, ...$, with $(z^{(1)}(0), z^{(1)}(-1))^\top = (0,0)^\top$ and $[\Sigma'(k-1)]_{11} = (1+\beta_{k-2})(\lambda_1 - c/k)$, $[\Sigma'(k-1)]_{12} = -\beta_{k-2}(\lambda_1 - c/k)$, $[\Sigma'(k-1)]_{21} = 1$, and $[\Sigma'(k-1)]_{22} = 0$. (Recall $\beta_k = k/(k+3)$ for $k = 0, 1, ...$ and $\beta_{-1} := 0$.) Now, take $c = \lambda_1/4$. Then, for $t \ge 0$, $\Sigma_1'(t) = \widehat{\Sigma}_1' - a_t'\Delta_1'$, with 1) $[\widehat{\Sigma}_1']_{11} = 2\lambda_1$, $[\widehat{\Sigma}_1']_{12} = -\lambda_1$, $[\widehat{\Sigma}_1']_{21} = 1$, $[\widehat{\Sigma}_1']_{22} = 0$; 2) $[\Delta_1']_{11} = -[\Delta_1']_{12} = \lambda_1$, and $[\Delta_1']_{21} = [\Delta_1']_{22} = 0$; and 3) $a_t' = \frac{3}{t+2} + \frac{1}{2(t+1)} - \frac{3}{4(t+2)(t+1)}$. Recall $\Sigma_i(t)$ in (2.41) and its representation in the paragraph below (2.43); $\Sigma_1'(t)$ has a very similar structure to $\Sigma_i(t)$ – the only difference is that a_t is replaced with a_t'. However, there still holds that $\|\Pi_{s=2}^{k-t+2}\Sigma_1'(k-s)\| \le \frac{8}{\sqrt{\lambda_1(1-\lambda_1)}}(\sqrt{\lambda_1})^{k-t}$, (just as with $\Sigma_i(t)$). This is because the key inequality (2.45) holds for all $a := a_t \in [0,2]$, and so it holds for all a_t', $t = 0, 1, ...$ Therefore, we can apply Theorem 2.7 to $\|w^{(1)}(k)\|$ using the following identification: $\widetilde{x}(k) \equiv z^{(1)}(k)$, $\widetilde{y}(k) \equiv w^{(1)}(k)$, $N \equiv 1$, $G \equiv \sqrt{2}\theta$, $\eta \equiv \lambda_1$, $\mu(W) \equiv \lambda_1$. Also, using $\log(z+1) \le z$, $z > 0$, upper bound $\mathcal{B}(r) = \sup_{z \ge 1/2} z r^z \log(z+1)$ as:

$$\mathcal{B}(r) \le \sup_{z \ge 1/2} z^2 r^z \le \frac{4}{e^2(-\log r)^2} \le \frac{4}{e^2(1-r)^2}\frac{(1+r)^2}{(1+r)^2} \le \frac{16}{e^2(1-r^2)^2},\; r \in (0,1).$$

Applying Theorem 2.7 to $\|w^{(1)}(k-1)\|$ with the above identifications and the bound on $\mathcal{B}(r)$:

$$\|w^{(1)}(k-1)\| \le \left(363\sqrt{2}\, c\theta\right)\left(\frac{1}{\sqrt{\lambda_1(1-\lambda_1)}}\frac{1}{(1-\lambda_1)^2}\right)\frac{1}{k-1},\; k \ge 2. \quad \text{(A.20)}$$

From (A.19) and (A.20):

$$\|z^{(1)}(k)\| \geq \frac{\sqrt{2}\,c\,\theta}{k}\left(1 - \frac{331\,k\,\sqrt{\lambda_1}}{(k-1)(1-\lambda_1)^{5/2}}\right) \geq \frac{\sqrt{2}\,c\,\theta}{2\,k},$$

for $k \geq 5$, and $\lambda_1 = 10^{-6}$. Now, from (2.63), and squaring the last inequality, we get that, for fixed $k \geq 5$, $M > 0$, $\max_{i=1,2}(f(x_i(k)) - f^\star) \geq M$ for $\theta(k, M) = \frac{2\,k\,\sqrt{M}}{c} = 8 \times 10^6\,k\,\sqrt{M}$. Thus, the result.

Remark on the distance to consensus. Consider $\widetilde{x}(k) = (I - J)x(k)$, $\widetilde{y}(k) = (I - J)y(k)$, and define $\widetilde{z}(k) = Q^\top z(k)$, $\widetilde{w}(k) = Q^\top w(k)$; $\widetilde{w}^{(1)}(k)$ and $\widetilde{z}^{(1)}(k)$ obey exactly the same equations (81) (replace in (81) $z^{(1)}(k)$ with $\widetilde{z}^{(1)}(k)$ and $w^{(1)}(k) = \widetilde{w}^{(1)}(k)$.) Thus, there holds that $\|\widetilde{x}(k)\| \geq \sqrt{2}c\theta/(2k)$.

A.5 Relaxing bounded gradients: Proof of (2.36) for the algorithm in [2]

We show that distributed gradient method in [2] has an arbitrarily slow worst case convergence rate when Assumption 2.3 is relaxed, while Assumptions 2.1 and 2.2 hold. We consider: the connected network with $N = 2$ nodes; the weight matrix with $W_{11} = W_{22} = 3/4$, $W_{12} = W_{21} = 1/4$; the candidate functions $f_i^\theta : \mathbb{R} \to \mathbb{R}$, $i = 1, 2$, as in (2.37), for $\theta > 0$; the initialization $x_1(0) = x_2(0) = 0$; and the algorithm step size $\alpha_k = \frac{c}{(k+1)^\tau}$, $c \in (0, 1/2]$, $\tau \geq 0$, $k = 0, 1, ...$ By evaluating the gradients of the f_i^θ's, and using the values of the entries of W, it is straightforward to show (by induction) that, $\forall k = 0, 1, ...$, $x_1(k) = -x_2(k) \geq 0$, with the update equation:

$$x_1(k) = (W_{11} - W_{12} - \alpha_{k-1})\,x_1(k-1) + \alpha_{k-1}\,\theta = \left(\frac{1}{2} - \alpha_{k-1}\right)x_1(k-1) + \alpha_{k-1}\,\theta,$$

and so $\|x_1(k)\| = x_1(k) \geq \alpha_{k-1}\,\theta$, $\forall k \geq 1$. Using the latter and (2.63), we obtain that the optimality gap $\max_{i=1,2}\,(f(x_i(k)) - f^\star) \geq \frac{\alpha_{k-1}^2\,\theta^2}{2}$. Now, it is clear that, for a fixed k, and any fixed step-size choice $c \in (0, 1/2]$, $\tau \geq 0$, the optimality gap $\max_{i=1,2}\,(f(x_i(k)) - f^\star) \geq \frac{\alpha_{k-1}^2\,\theta^2}{2} = \frac{c^2\,\theta^2}{2\,k^{2\,\tau}}$ can be made arbitrarily large by choosing a sufficiently large θ.

Appendix B

Tecnical Proofs for Chapter 3

B.1 Proofs of Lemmas 3.3 and 3.13

Proof: [Proof of Lemma 3.3] We first prove (3.3). For $t = k-1$, $\widetilde{\Phi}(k,t) = I$ and (3.3) holds. Next, fix some t, $0 \leq t \leq k-2$. We use the following inequality for a $N \times N$ matrix A: $\|A\|^2 \leq N \sum_{i=1}^{N} \|A\, e_i\|^2$, where e_i is the i-th canonical vector ($A\, e_i$ is the i-th column of A.) Using the latter with $A \equiv \widetilde{\Phi}(k,t)$, and taking expectation:

$$\mathbb{E}\left[\left\|\widetilde{\Phi}(k,t)^\top \widetilde{\Phi}(k,t)\right\|\right] = \mathbb{E}\left[\left\|\widetilde{\Phi}(k,t)\right\|^2\right] \leq N \sum_{i=1}^{N} \mathbb{E}\left[\left\|\widetilde{\Phi}(k,t)\, e_i\right\|^2\right]. \tag{B.1}$$

Denote by $\chi_i(s+1) := \widetilde{\Phi}(s,t) = \widetilde{W}(s+t+2)...\widetilde{W}(t+2)\, e_i$, $s = 0, ..., k-t-2$, and $\chi_i(0) = e_i$. The vectors $\chi_i(s)$ are recursively expressed as:

$$\chi_i(s+1) = \widetilde{W}(s+t+2)\chi_i(s), \;\; s = 0, ..., k-t-2, \;\; \chi_i(0) = e_i.$$

Using the above, independence of the matrices $\widetilde{W}(k)$'s, and nesting expectations:

$$\begin{aligned}
\mathbb{E}\left[\|\chi_i(k-t-1)\|^2\right] &= \mathbb{E}\left[\mathbb{E}\left[\chi_i(k-t-2)^\top \widetilde{W}(k)^2 \chi_i(k-t-2) \,|\, \widetilde{W}(k-1), ..., \widetilde{W}(t+2)\right]\right] \\
&= \mathbb{E}\left[\chi_i(k-t-2)^\top \mathbb{E}\left[\widetilde{W}(k)^2\right] \chi_i(k-t-2)\right] \\
&\leq \mathbb{E}\left[\left\|\mathbb{E}\left[\widetilde{W}(k)^2\right]\right\| \, \|\chi_i(k-t-2)\|^2\right] \\
&\leq \overline{\mu}^2\, \mathbb{E}\left[\|\chi_i(k-t-2)\|^2\right].
\end{aligned}$$

Applying the same argument to $E\left[\|\chi_i(k-t-2)\|^2\right]$, and continuing successively, obtain, for all i:

$$\mathbb{E}\left[\|\widetilde{\Phi}(k,t)\,e_i\|^2\right] = \mathbb{E}\left[\|\chi_i(k-t-1)\|^2\right] \leq (\overline{\mu}^2)^{k-t-1}\,\|e_i\|^2 = (\overline{\mu}^2)^{k-t-1}.$$

Plugging the latter bound in (B.1), the result in (3.3) follows. Next, (3.2) follows from (3.3) and Jensen's inequality. It remains to prove (3.4). We first consider $0 \leq s < t \leq k-2$. (The case $t < s$ is symmetric.) Using the independence of the $\widetilde{W}(k)$'s, the sub-multiplicative property of norms, and taking expectation, obtain:

$$\begin{aligned}\mathbb{E}\left[\left\|\widetilde{\Phi}(k,s)^\top\widetilde{\Phi}(k,t)\right\|\right] &\leq \mathbb{E}\left[\|\widetilde{W}(t+1)...\widetilde{W}(s+2)\|\right]\mathbb{E}\left[\left\|\widetilde{\Phi}(k,t)\right\|^2\right] \\ &\leq \left(N\,\overline{\mu}^{t-s}\right)\left(N^2\,(\overline{\mu}^2)^{k-t-2}\right) = N^3\,\overline{\mu}^{(k-t-1)+(k-s-1)}. \qquad \text{(B.2)}\end{aligned}$$

Inequality (B.2), we applied (3.2) and (3.3). thus, the result in (3.4) for $s,t \in \{0,...,k-2\}$. Now, if $s = k-1$, $t < k-1$, $\widetilde{\Phi}(k,s)^\top\widetilde{\Phi}(k,t) = \widetilde{\Phi}(k,t)$ and the result reduces to (3.3). The case $s < k-1$, $t = k-1$ is symmetric. Finally, if $s = k-1, t = k-1$, the result is trivial. The proof is complete. □

Proof: [Proof of Lemma 3.13] We first prove (3.43). By definition of $\mathcal{W}(k)$ in (3.42), it is the product of τ_k i.i.d. matrices $W(t)$ that obey Assumptions 3.1 and 3.2. Hence, by (3.3), we have:

$$\mathbb{E}\left[\|\mathcal{W}(k)\|^2\right] \leq (\overline{\mu}^2)^{\tau_k} = N^2\,e^{2\tau_k\,\log(\overline{\mu})} \leq N^2\,e^{-2(3\log k+\log N)} = \frac{1}{k^6},$$

and we obtain (3.43).

We now prove (3.45). Consider $\widetilde{\Psi}(k,t) := \widetilde{\mathcal{W}}(k),...,\widetilde{\mathcal{W}}(t+1)$, $k \geq t+1$. We use the following inequality for square matrices A and B: $\|B^\top A^\top AB\| \leq \|A^\top A\|\,\|B^\top B\| = \|B\|^2\,\|A\|^2$. Applying the latter successively $k-t$ times, we obtain:

$$\left\|\widetilde{\Psi}(k,t)^\top\widetilde{\Psi}(k,t)\right\| \leq \left\|\widetilde{\mathcal{W}}(k)\right\|^2 ... \left\|\widetilde{\mathcal{W}}(t+1)\right\|^2.$$

Using the independence, taking expectation, and applying (3.43), we obtain (3.45). Now, (3.44) follows by Jensen's inequality from (3.45); relation (3.46) can be proved similarly and the details are omitted for brevity. □

B.2 Proof of (3.61)–(3.62)

Recall the random graph $G(k)$. Further, recall that we impose here, for a certain connected graph G_0, that $G(k) = G_0$ with a positive probability p_G. The latter Assumption, together with the fact that $\underline{w}$ in Assumption 3.1 is bounded away from zero, implies that there exists a scalar $\mu_4 \in [0, 1)$, such that:

$$\mathbb{E}\left[\left\|\widetilde{W}(k)\right\|^4\right] \leq (\mu_4)^4.$$

In particular, μ_4 can be taken as:

$$\mu_4 = (1 - p_G) + p_G\left(1 - \underline{w}^2\,\lambda_{\rm F}\left(G_0\right)\right)^2 < 1,$$

where $\lambda_{\rm F}\left(G_0\right)$ is the second largest eigenvalue (algebraic connectivity) of the (unweighted) Laplacian matrix of the graph G_0. Next, consider (3.38). Denote by $\widetilde{f}_k := \frac{1}{N}(f(\overline{x}(k)) - f^\star)$. Squaring (3.38), taking expectation, and using the Cauchi-Schwarz inequality, obtain:

$$\begin{aligned}\mathbb{E}\left[(\widetilde{f}_k)^2\right] \;\leq\; & \frac{4R^2}{c^2\,k^2} + \frac{4RL}{N\,c}\,\frac{1}{k^2}\,\sum_{t=1}^{k}\frac{(t+1)^2}{t}\mathbb{E}\left[\|\widetilde{y}(t-1)\|^2\right] \\ & + \;\frac{L^2}{N^2\,k^2}\,\sum_{t=1}^{k}\sum_{s=1}^{k}\frac{(t+1)^2}{t}\frac{(s+1)^2}{s}\left(\mathbb{E}\left[\,\|\widetilde{y}(t-1)\|^4\,\right]\right)^{1/2}\left(\mathbb{E}\left[\,\|\widetilde{y}(s-1)\|^4\,\right]\right)^{1/2},\end{aligned} \tag{B.3}$$

where we recall $R := \|\overline{x}(0) - x^\star\|$. The first term in (B.3) is clearly $O(1/k^2)$; the second term is, applying Theorem 3.7, $O(\log k/k^2)$. It remains to upper bound the third term, which requires an upper bound on the fourth moment of $\|\widetilde{y}(t)\|$, for all $t = 0, 1, ..., k-1$. Recall (3.32), and denote by $\mathcal{U}_k := \|\widetilde{W}(k)\|$. Fix $s < t$, $s, t \in \{0, 1, ..., k-1\}$. By the sub-multiplicative property of norms:

$$\left\|\widetilde{\Phi}(k,t)^\top\widetilde{\Phi}(k,s)\right\| \leq \left(\mathcal{U}_k^2\,\mathcal{U}_{k-1}^2\,...\,\mathcal{U}_{t+2}^2\right)\,\left(\mathcal{U}_{t+1}\,...\,\mathcal{U}_{s+2}\right).$$

Similarly, for $t = s$:

$$\left\|\widetilde{\Phi}(k,t)^\top\widetilde{\Phi}(k,t)\right\| \leq \mathcal{U}_k^2\,\mathcal{U}_{k-1}^2\,...\,\mathcal{U}_{t+2}^2.$$

Denote by:

$$\widehat{\mathcal{U}}(t,s) := \mathcal{U}_t\,\mathcal{U}_{t-1}\,...\,\mathcal{U}_{s+1},$$

for $t > s$, and $\widehat{\mathcal{U}}(t,t) = I$. Further, squaring (3.32):

$$\begin{aligned} \|\widetilde{z}(k)\|^4 &\leq (9c^4N^2G^4) \sum_{t_1=0}^{k-1}\sum_{t_2=0}^{k-1}\sum_{t_3=0}^{k-1}\sum_{t_4=0}^{k-1} \widehat{b}(k,t_1)\,\widehat{b}(k,t_2)\,\widehat{b}(k,t_3)\,\widehat{b}(k,t_4) \\ &\times \left(\widehat{\mathcal{U}}(k,t_1+1)\right)^4 \left(\widehat{\mathcal{U}}(t_1+1,t_2+1)\right)^3 \left(\widehat{\mathcal{U}}(t_2+1,t_3+1)\right)^2 \left(\widehat{\mathcal{U}}(t_3+1,t_4+1)\right)^4 \\ &\times \frac{1}{(t_1+1)(t_2+1)(t_3+1)(t_4+1)}, \end{aligned} \tag{B.4}$$

where $\widehat{b}(k,t) := \frac{8(k-t-1)(t+1)}{k} + 5$. Fix $t_i \in \{0,1,...,k-1\}$, $i = 1,...,4$, with $t_1 \geq t_2 \geq t_3 \geq t_4$. Using independence of the $\mathcal{U}_k$'s, and the following inequality: $\left(\mathbb{E}\left[\mathcal{U}_k^j\right]\right)^{4/j} \leq \mathbb{E}\left[\mathcal{U}_k^4\right]$, $j = 1,2,3$, we obtain:

$$\begin{aligned} &\mathbb{E}\left[\,(\widehat{\mathcal{U}}(k,t_1+1))^4\,(\widehat{\mathcal{U}}(t_1+1,t_2+1))^3\,(\widehat{\mathcal{U}}(t_2+1,t_3+1))^2\,(\widehat{\mathcal{U}}(t_3+1,t_4+2))\,\right] \\ \leq\;& (\mu_4^4)^{k-t_1-1}\,(\mu_4^3)^{t_1-t_2}\,(\mu_4^2)^{t_2-t_3}\,(\mu_4)^{t_3-t_4} \\ =\;& (\mu_4)^{\sum_{i=1}^4 (k-t_i)-1}. \end{aligned} \tag{B.5}$$

Taking expectation in (B.4), and applying (B.5), we get:

$$\mathbb{E}\left[\|\widetilde{z}(k)\|^4\right] \leq (9c^4N^2G^4) \left(\sum_{t=0}^{k-1} \widehat{b}(k,t)\,\mu_4^{k-t}(t+1)^{-1}\right)^4 = O(1/k^4), \tag{B.6}$$

where the last equality uses Lemma 3.9. Applying the last bound to (B.3), we conclude that the third term in (B.3) is $O(\log^2 k/k^2)$. Thus, overall, $\mathbb{E}\left[(\widetilde{f}_k)^2\right] = O(\log^2 k/k^2)$. Next, express $f(x_i(k)) - f^\star = N\widetilde{f}_k + (f(x_i(k)) - f(\overline{x}(k)))$, and use $(f(x_i(k)) - f^\star)^2 \leq 2(N\widetilde{f}_k)^2 + 2(f(x_i(k)) - f^\star)^2 \leq 2(N\widetilde{f}_k)^2 + 2GN\|\widetilde{x}(k)\|^2$. Taking expectation in the last inequality, applying Theorem 3.7, and using (B.6), the result (3.61) follows.

Consider now mD–NC. We modify the value τ_k here to equal $\tau_k = \lceil \frac{3\log k}{-\log \mu_4} \rceil$. Now, result (3.62) can be proved similarly to (3.61), and details are omitted for brevity.

Appendix C

Proofs and Technical Details for Chapter 5

C.1 Proof of Lemma 5.1 (a sketch)

Eqn. (5.18) follows from the expectation of (5.3). To prove the remaining of the Lemma, we find W^2, $\overline{W}^2$, and the expectation W^2. We obtain successively:

$$\begin{aligned}
W^2 &= (\, C \odot \mathcal{A} + I - \mathrm{diag}(C\,\mathcal{A})\,)^2 \\
&= (C \odot \mathcal{A})^2 + \mathrm{diag}^2(C\,\mathcal{A}) + I + 2\,C \odot \mathcal{A} - 2\,\mathrm{diag}(C\,\mathcal{A}) - (C \odot \mathcal{A})\,\mathrm{diag}(C\,\mathcal{A}) \\
&- \mathrm{diag}(C\,\mathcal{A})\,(C \odot \mathcal{A}) \\
\overline{W}^2 &= (C \odot P)^2 + \mathrm{diag}^2(C\,P) + I + 2\,C \odot P - 2\,\mathrm{diag}\,(C\,P) \\
&\quad - [(C \odot P)\,\mathrm{diag}\,(C\,P) + \mathrm{diag}\,(C\,P)\,(C \odot P)] \\
\mathbb{E}\left[W^2\right] &= \mathbb{E}\left[(C \odot \mathcal{A})^2\right] + \mathbb{E}\left[\mathrm{diag}^2(C\,\mathcal{A})\right] + I + 2\,C \odot P \\
&\quad -2\,\mathrm{diag}\,(C\,P) - \mathbb{E}[\,(C \odot \mathcal{A})\,\mathrm{diag}(C\,\mathcal{A}) + \mathrm{diag}(C\,\mathcal{A})\,(C \odot \mathcal{A})\,]
\end{aligned}$$

We will next show the following three equalities:

$$\mathbb{E}\left[(C \odot \mathcal{A})^2\right] = (C \odot P)^2 + \widehat{C}^T \left\{R_A \odot (11^T \otimes I)\right\} \widehat{C} \tag{C.1}$$

$$\mathbb{E}\left[\mathrm{diag}^2\,(C\mathcal{A})\right] = \mathrm{diag}^2(C\,P) + \widehat{C}^T \left\{R_A \odot (I \otimes 11^T)\right\} \widehat{C} \tag{C.2}$$

$$\mathbb{E}\left[(C \odot \mathcal{A})\mathrm{diag}\,(C\mathcal{A}) + \mathrm{diag}\,(C\mathcal{A})\,(C \odot \mathcal{A})\right] = \tag{C.3}$$

$$(C \odot P)\mathrm{diag}\,(CP) + \mathrm{diag}\,(CP)\,(C \odot P) - \widehat{C}^T \left\{R_A \odot B\right\} \widehat{C}$$

First, consider (C.1) and find $\mathbb{E}\left[(C \odot \mathcal{A})^2\right]$. Algebraic manipulations allow to write $(C \odot \mathcal{A})^2$ as follows:

$$(C \odot \mathcal{A})^2 = \widehat{C}^T \left\{\mathcal{A}_2 \odot (11^T \otimes I)\right\} \widehat{C}, \;\; \mathcal{A}_2 = \mathrm{Vec}(\mathcal{A})\mathrm{Vec}^T(\mathcal{A}) \tag{C.4}$$

To compute the expectation of (C.4), we need $\mathbb{E}[\mathcal{A}_2]$ that can be written as

$$\mathbb{E}[\mathcal{A}_2] = P_2 + R_A, \;\; \text{with} \;\; P_2 = \;\mathrm{Vec}(P)\;\mathrm{Vec}^T(P).$$

Equation (C.1) follows, realizing that

$$\widehat{C}^T \left\{P_2 \odot (11^T \otimes I)\right\} \widehat{C} = (C \odot P)^2.$$

Now consider (C.2) and (C.3). After algebraic manipulations, it can be shown that

$$\begin{aligned} \mathrm{diag}^2(C\,\mathcal{A}) &= \widehat{C}^T \left\{\mathcal{A}_2 \odot (I \otimes 11^T)\right\} \widehat{C} \\ (C \odot \mathcal{A})\;\mathrm{diag}(C\,\mathcal{A}) \;+\; \mathrm{diag}(C\,\mathcal{A})\;(C \odot \mathcal{A}) &= \widehat{C}^T \{\mathcal{A}_2 \odot B\} \widehat{C} \end{aligned}$$

Computing the expectations in the last two equations leads to eqn. (C.2) and eqn. (C.3).

Using equalities (C.1), (C.2), and (C.3) and comparing the expressions for $\overline{W}^2$ and $\mathbb{E}[W^2]$ leads to:

$$R_C = \mathbb{E}[W^2] - \overline{W}^2 = \widehat{C}^T \;\{\; R_A \odot (I \otimes 11^T + 11^T \otimes I - B)\}\; \widehat{C} \tag{C.5}$$

This completes the proof of Lemma 5.1.

C.2 Subgradient step calculation for the case of spatially correlated links

To compute the subgradient H, from eqns. (5.44) and (5.45) we consider the computation of $\mathbb{E}\left[W^2 - J\right] = \overline{W}^2 - J + R_C$. Matrix $\overline{W}^2 - J$ is computed in the same way as for the uncorrelated case. To compute R_C, from (C.5), partition the matrix R_A into $N \times N$ blocks:

$$R_A = \begin{pmatrix} R_{11} & R_{12} & \dots & R_{1N} \\ R_{21} & R_{22} & \dots & R_{2N} \\ \vdots & \dots & \dots & \vdots \\ R_{N1} & R_{N2} & \dots & R_{NN} \end{pmatrix}$$

Denote by d_{ij}, by c_{ij}^l, and by r_{ij}^l the diagonal, the l-th column, and the l-th row of the block R_{ij}. It can be shown that the matrix R_C can be computed as follows:

$$\begin{aligned}
[R_C]_{ij} &= W_i^T \left(d_{ij} \odot W_j\right) - W_{ij} \left(W_i^T c_{ij}^i + W_j^T r_{ij}^j\right), \; i \neq j \\
[R_C]_{ii} &= W_i^T \left(d_{ii} \odot W_i\right) + W_i^T R_{ii} W_i
\end{aligned}$$

Denote by $R_A(:,k)$ the k-th column of the matrix R_A and by

$$\begin{aligned}
k_1 &= \left(e_j^T \otimes I_N\right) R_A(:,(i-1)N+j), \; k_2 = \left(e_i^T \otimes I_N\right) R_A(:,(j-1)N+i), \\
k_3 &= \left(e_i^T \otimes I_N\right) R_A(:,(i-1)N+j), \; k_4 = \left(e_j^T \otimes I_N\right) R_A(:,(j-1)N+i).
\end{aligned}$$

Quantities k_1, k_2, k_3 and k_4 depend on (i,j) but for the sake of the notation simplicity indexes are omitted. It can be shown that the computation of H_{ij}, $(i,j) \in E$ boils down to:

$$H_{ij} = 2\,u_i^2\,W_i^T\,c_{ii}^j + 2\,u_j^2\,W_j^T\,c_{jj}^i + 2\,u_i\,W_j^T\,(u \odot k_1) \; + \; 2\,u_j\,W_i^T\,(u \odot k_2) \; - 2\,u_i\,u_j\,W_j^T\,c_{ji}^j -$$

$$2\,u_i\,u_j\,W_i^T\,c_{ij}^i - 2\,u_i\,W_i^T\,(u \odot k_3) \; - 2\,u_j\,W_j^T\,(u \odot k_4) + 2\,P_{ij}\,(u_i - u_j)\,u^T \left(\overline{W}_j \; - \; \overline{W}_i\right)$$

C.3 Numerical optimization for the broadcast gossip algorithm

With broadcast gossip, the matrix $W(k)$ can take N different realizations, corresponding to the broadcast cycles of each of the N sensors. We denote these realizations by $W^{(i)}$, where i indexes the broadcasting node. We can write the random realization of the broadcast gossip matrix $W^{(i)}$, $i = 1, ..., N$, as follows:

$$W^{(i)}(k) = C \odot \mathcal{A}^{(i)}(k) + I - \text{diag}\left(C\,\mathcal{A}^{(i)}(k)\right), \tag{C.6}$$

where $\mathcal{A}_{li}^{(i)}(k) = 1$, if $l \in \Omega_i$. Other entries of $\mathcal{A}^{(i)}(k)$ are zero.

Similarly in Appendix A, we can arrive at the expressions for $\mathbb{E}\left[W^T W\right] := \mathbb{E}\left[W^T(k) W(k)\right]$ and for $\mathbb{E}\left[W^T J W\right] := \mathbb{E}\left[W^T(k) J W(k)\right]$, for all k. We remark that the matrix W needs not to be symmetric for

the broadcast gossip and that $W_{ij} = 0$, if $(i,j) \notin E$.

$$\begin{aligned}
\mathbb{E}\left[\left(W^T W\right)_{ii}\right] &= \frac{1}{N}\sum_{l=1,l\neq i}^{N} W_{li}^2 + \frac{1}{N}\sum_{l=1,l\neq i}^{N}(1-W_{il})^2 \\
\mathbb{E}\left[\left(W^T W\right)_{ij}\right] &= \frac{1}{N}W_{ij}(1-W_{ij}) + \frac{1}{N}W_{ji}(1-W_{ji}),\ i\neq j \\
\mathbb{E}\left[\left(W^T JW\right)_{ii}\right] &= \frac{1}{N^2}\left(1+\sum_{l\neq i}W_{li}\right)^2 + \frac{1}{N^2}\sum_{l=1,l\neq i}^{N}(1-W_{il})^2 \\
\left[\mathbb{E}\left[W^T JW\right)_{ij}\right] &= \frac{1}{N^2}(1-W_{ji})(1+\sum_{l=1,l\neq i}^{N}W_{li}) + \frac{1}{N^2}(1-W_{ij})(1+\sum_{l=1,l\neq j}^{N}W_{lj}) \\
&+ \frac{1}{N^2}\sum_{l=1,l\neq i,l\neq j}^{N}(1-W_{il})(1-W_{jl}),\ i\neq j
\end{aligned}$$

Denote by $W^{\mathrm{BG}} := \mathbb{E}\left[W^T W\right] - \mathbb{E}\left[W^T JW\right]$ and recall the definition of the MSdev rate $\psi(W)$ (5.47). We have that $\psi(W) = \lambda_{\max}\left(W^{\mathrm{BG}}\right)$. We proceed with the calculation of a subgradient of $\psi(W)$. The partial derivative of the cost function $\psi(W)$ with respect to weight $W_{i,j}$ is given by:

$$\frac{\partial}{\partial W_{i,j}}\lambda_{\max}\left(W^{\mathrm{BG}}\right) = q^T\left(\frac{\partial}{\partial W_{i,j}}W^{\mathrm{BG}}\right)q$$

where q is eigenvector associated with the maximal eigenvalue of the matrix W^{BG}. Finally, partial derivatives of the entries of the matrix W^{BG} with respect to weight $W_{i,j}$ are given by the following set of equations:

$$\begin{aligned}
\frac{\partial}{\partial W_{i,j}}W_{i,i}^{\mathrm{BG}} &= -2\frac{N-1}{N}(1-W_{i,j}) \\
\frac{\partial}{\partial W_{i,j}}W_{j,j}^{\mathrm{BG}} &= \frac{2}{N}W_{i,j} - \frac{2}{N}(1-\sum_{l=1,l\neq j}^{N}W_{l,j}) \\
\frac{\partial}{\partial W_{i,j}}W_{i,j}^{\mathrm{BG}} &= \frac{1}{N}(1-2W_{i,j}) - \frac{1}{N^2}(-1-\sum_{l=1,l\neq j}^{N}W_{l,j} - W_{i,j}) \\
\frac{\partial}{\partial W_{i,j}}W_{i,l}^{\mathrm{BG}} &= \frac{1}{N^2}(1-W_{l,j}),\ l\neq i, l\neq j \\
\frac{\partial}{\partial W_{i,j}}W_{i,j}^{\mathrm{BG}} &= -\frac{1}{N^2}(1-W_{l,j}),\ l\neq i, l\neq j \\
\frac{\partial}{\partial W_{l,m}}W_{i,j}^{\mathrm{BG}} &= 0,\ otherwise.
\end{aligned}$$

Appendix D

Technical Proofs for Chapter 6

D.1 Proof of Lemma 6.8

We first need a standard result from topology (proof omitted for brevity.)

Lemma D.1 Let $\mathcal{X}$ and $\mathcal{Y}$ be topological spaces, where $\mathcal{Y}$ is compact. Suppose the function: $\kappa : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is continuous (with respect to the product topology on $\mathcal{X} \times \mathcal{Y}$ and the usual topology on $\mathbb{R}$; $\times$ denotes Cartesian product.) Then, the function $\gamma : \mathcal{X} \to \mathbb{R}$, $\gamma(a) := \inf\{\kappa(a, b) : b \in \mathcal{Y}\}$ is continuous.

Proof: [Proof of Lemma 6.8] Denote by $\mathcal{P}_i : \mathbb{R}^{m(N+2M)} \to \mathbb{R}^m$ the projection map $\mathcal{P}_i(z) = z_i$, $i = 1, ..., N + 2M$. Further, denote by $\mathcal{P}_i(\Gamma(\epsilon)) := \{z_i \in \mathbb{R}^m : z_i = \mathcal{P}_i(z), \text{ for some } z \in \Gamma(\epsilon)\}$. The set $\mathcal{P}_i(\Gamma(\epsilon))$ is compact, for all $i = 1, ..., N + 2M$, because the set $\Gamma(\epsilon)$ is compact. Consider now the set $\mathbb{R}^{m(N+2M)} \supset C_\epsilon := \mathcal{P}_1(\Gamma(\epsilon)) \times \mathcal{P}_2(\Gamma(\epsilon)) \times ... \times \mathcal{P}_{N+2M}(\Gamma(\epsilon))$, where the symbol $\times$ denotes the Cartesian product of the sets. Clearly, $C_\epsilon \supset \Gamma_\epsilon(B)$. We will show that L^i is continuous on C_ϵ, i.e., that $L^i : C_\epsilon \to \mathbb{R}$ is continuous, which will imply the claim of Lemma 6.8. Recall the definition of L^i in eqn. (6.18). It is easy to see that the minimum in eqn. (6.18) is attained on the set $\mathcal{P}_i(\Gamma(\epsilon))$, i.e., that $L^i(z) = \min_{w_i \in \mathcal{P}_i(\Gamma(\epsilon))} L(z_1, z_2, ..., z_{i-1}, w_i, z_{i+1}, ..., z_{N+2M})$. Thus, by Lemma 6.12, and because the function $L : \mathbb{R}^{m(N+2M)} \to \mathbb{R}$ is continuous, the function $L^i : \mathcal{P}_1(\Gamma(\epsilon)) \times ... \times \mathcal{P}_{i-1}(\Gamma(\epsilon)) \times \mathcal{P}_{i+1}(\Gamma(\epsilon)) \times ... \times \mathcal{P}_{N+2M}(\Gamma(\epsilon)) \to \mathbb{R}$ is continuous. But this means that $L^i : C_\epsilon \to \mathbb{R}$ is also continuous. □

D.2 Convergence proof of the P–AL–MG algorithm

We first introduce an abstract model of the P–AL–MG algorithm. First, we impose an additional assumptions that the link failures are spatially independent, i.e., the Bernoulli states $A_{ij}(k)$ and $A_{lm}(k)$ of different

links at time slot k are independent. Define the sets $Y(\Omega_i) := \{y_{ji} : \ j \in \Omega_i\}$ and the class $Y(O_i) := \{y_{ji} : \ j \in O_i\}$, where $O_i \subset \Omega_i$. One distinct set $Y(O_i)$ is assigned to each distinct subset O_i of Ω_i. (Clearly, $Y(\Omega_i)$ belongs to a class of sets $Y(O_i)$, as Ω_i is a subset of itself.) With P–AL–MG, at iteration k, minimization is performed either with respect to x_i, $i \in \{1, ..., N\}$, or with respect to some $Y(O_i)$. If none of the neighbors of node i receives successfully a message, then iteration k is void. Define the following collection of the subsets of primal variables: $\Pi := \{\{x_1\}, ..., \{x_N\}, Y(\Omega_1), ..., Y(\Omega_N)\}$. Collection Π constitutes a partition of the set of primal variables; that is, different subsets in Π are disjoint and their union contains all primal variables. Further, denote each of the subsets $\{x_i\}$, $Y(O_i)$, $Y(\Omega_i)$, with appropriately indexed Z_s, $s = 1, ..., S$. Then, with P–AL–MG, at time slot k, $L(z)$ is optimized with respect to one Z_s, $s = 1, ..., S$. Define $\xi(k)$, $k = 1, 2, ...$, as follows: $\xi(k) = s$, if, at time slot k, $L(z)$ is optimized with respect to Z_s; $\xi(k) = 0$, if, at k, no variable gets updated–when all transmissions at time slot k are unsuccessful. Denote by $P(Z_s) = \text{Prob}\left(\xi(k) = s\right)$. Under spatial independence of link failures, $P(Z_s)$ can be shown to be strictly positive for all s. It can be shown that $\xi(k)$ are i.i.d. Consider now (6.16) and P–AL–MG. All results for P–AL–G remain valid for P–AL–MG also–only the expressions for the expected decrease of $L(\cdot)$ per iteration, $\psi(z)$, (Lemma 6.7), and the proof of Lemma 6.8 change. Denote by $L^{(Z_s)}(z)$ the optimal value after minimizing $L(\cdot)$ with respect to Z_s at point z (with the other blocks z_j, $z_j \notin Z_s$, fixed.) Then: $\psi(z) = \sum_{s=1}^{S} P(Z_s) \left(L^{(Z_s)} - L(z)\right)$. Recall $\phi(z) = -\psi(z)$ and the set $\Gamma(\epsilon)$, for some $\epsilon > 0$. Lemma 6.8 remains valid for P–AL–MG. To see this, first remark that $\phi(z) \geq 0$, for all $z \in F$. We want to show that $\phi(z) > 0$, for all $z \in \Gamma(\epsilon)$. Suppose not. Then, $L(z) = L^{(Z_s)}(z)$, for all Z_s, $s = 1, ..., S$. Then, in particular, $L(z) = L^{(Z_s)}(z)$, for all Z_s in the partition Π. Because $P(Z_s) > 0, \forall s$, this implies that the point z is block-optimal (where now, in view of Definition 6.2, Z_s is considered a single block). By Assumption 3, z is also optimal, which contradicts $z \in \Gamma(\epsilon)$. Thus, $\phi(z) > 0$ for all $z \in \Gamma(\epsilon)$. The proof now proceeds as with the proof of Lemma 6.8 for algorithm P–AL–G.

D.3 Convergence proof of the P–AL–BG algorithm

P–AL–BG is completely equivalent to P–AL–G, from the optimization point of view. P–AL–BG can be modeled in the same way as in Alg. 2, with a difference that, with P–AL–BG: 1) there is a smaller number ($= N$) of primal variables: $z_i := x_i$, $i = 1, ..., N$; and 2) $\text{Prob}(\zeta(k) = 0) = 0$. Thus, the analysis in section V is also valid for P–AL–BG.

Bibliography

[1] T. Aysal, M. Yildiz, A. Sarwate, and A. Scaglione, "Broadcast gossip algorithms for consensus," *IEEE Transactions on Signal Processing*, vol. 57, pp. 2748–2761, July 2009.

[2] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, pp. 48–61, January 2009.

[3] D. Jakovetic, J. M. F. Moura, and J. Xavier, "Distributed detection over noisy networks: Large deviations analysis," *IEEE Transactions on Signal Processing*, vol. 60, pp. 4306–4320, August 2012.

[4] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *IPSN 2004, 3rd International Symposium on Information Processing in Sensor Networks*, (Berkeley, California, USA), pp. 20 – 27, April 2004.

[5] J. A. Bazerque and G. B. Giannakis, "Distributed spectrum sensing for cognitive radio networks by exploiting sparsity," *IEEE Transactions on Signal Processing*, vol. 58, pp. 1847–1862, March 2010.

[6] P. D. Lorenzo and S. Barbarossa, "A bio-inspired swarming algorithm for decentralized access in cognitive radio," *IEEE Transactions on Signal Processing*, vol. 59, pp. 6160–6174, December 2011.

[7] X. Zhao and A. H. Sayed, "Diffusion adaptation over networks under imperfect information exchange and non-stationary data," *IEEE Trans. Sig. Process.*, vol. 60, pp. 3460–3475, Julyfi 2012.

[8] G. Mateos, J. A. Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Transactions on Signal Processing*, vol. 58, pp. 5262–5276, November 2010.

[9] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Fast distributed gradient methods," *conditionally accepted to IEEE Trans. Autom. Contr.*, Jan. 2013. initial submission, Nov. 2011; available at: http://arxiv.org/abs/1112.2972.

[10] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Transactions on Information Theory*, vol. 52, pp. 2508–2530, June 2006.

[11] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate $O(1/k^2)$," *Dokl. Akad. Nauk SSSR*, vol. 269, pp. 543–547, 1983. (in Russian).

[12] J. Duchi, A. Agarwal, and M. Wainwright, "Dual averaging for distributed optimization: Convergence and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, pp. 592–606, March 2012.

[13] S. Kar and J. M. F. Moura, "Sensor networks with random links: Topology design for distributed consensus," *IEEE Transactions on Signal Processing*, vol. 56, pp. 3315–3326, July 2008.

[14] P. Tseng, "Convergence of block coordinate descent method for nondifferentiable minimization," *J. Optim. Theory Appl.*, vol. 109, pp. 475–494, June 2001.

[15] D. Bersekas, *Nonlinear Programming*. Athena Scientific, 1995.

[16] J. N. Tsitsiklis, *Problems in decentralized decision making and computation*. Ph.d., MIT, Cambridge, MA, 1984.

[17] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Control*, vol. 31, pp. 803–812, September 1986.

[18] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Trans. Sig. Process.*, vol. 60, pp. 4289–4305, Aug. 2012.

[19] C. Lopes and A. H. Sayed, "Adaptive estimation algorithms over distributed networks," in *21st IEICE Signal Processing Symposium*, (Kyoto, Japan), Nov. 2006.

[20] S. S. Ram, A. Nedic, and V. Veeravalli, "Asynchronous gossip algorithms for stochastic optimization," in *CDC '09, 48th IEEE International Conference on Decision and Control*, (Shanghai, China), pp. 3581 – 3586, December 2009.

[21] A. Nedic, A. Ozdaglar, and A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, pp. 922–938, April 2010.

[22] S. Ram, A. Nedic, and V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *Journal of Optimization Theory and Applications*, vol. 147, no. 3, pp. 516–545, 2011.

[23] I. Lobel, A. Ozdaglar, and D. Feijer, "Distributed multi-agent optimization with state-dependent communication," *Mathematical Programming*, vol. 129, no. 2, pp. 255–284, 2011.

[24] I. Lobel and A. Ozdaglar, "Convergence analysis of distributed subgradient methods over random networks," in *46th Annual Allerton Conference onCommunication, Control, and Computing*, (Monticello, Illinois), pp. 353 – 360, September 2008.

[25] B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, "Subgradient methods and consensus algorithms for solving separable distributed control problems," in *CDC'08, 47th IEEE Conference on Decision and Control*, (Cancun, Mexico), pp. 4185–4190, December 2008.

[26] I. Matei and J. S. Baras, "Performance evaluation of the consensus-based distributed subgradient method under random communication topologies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 754–771, 2011.

[27] K. Tsianos and M. Rabbat, "Distributed consensus and optimization under communication delays," in *to apper in proc. 49th Allerton Conference on Communication, Control, and Computing*, (Monticello, Illinois), pp. 974–982, Sept. 2011.

[28] M. Zhu and S. Martínez, "On distributed convex optimization under inequality and equality constraints," *IEEE Transactions on Automatic Control*, vol. 57, pp. 151–164, Jan. 2012.

[29] I.-A. Chen and A. Ozdaglar, "A fast distributed proximal gradient method," in *Allerton Conference on Communication, Control and Computing*, (Monticello, IL), October 2012.

[30] I. A. Chen, "Fast distributed first-order methods," *Master thesis, Massachusetts Institute of Technology*, 2012.

[31] Z. Q. Luo and P. Tseng, "On the linear convergence of descent methods for convex essentially smooth optimization," *Siam J. Control and Optimization*, vol. 30, no. 2, pp. 408–425, 1992.

[32] B. W. Kort and D. P. Bertsekas, "Combined primal-dual and penalty methods for convex programming," *Siam J. Control and Optimization*, vol. 14, pp. 268–294, Feb. 1976.

[33] R. T. Rockafellar, "Augmented Lagrangian and applications of the proximal point algorithm in convex programming," *Math. Oper. Res.*, vol. 1, pp. 97–116, 1976.

[34] M. Hong and Z.-Q. Luo, "On the linear convergence of the alternating direction method of multipliers," 2012. arxiv post: arxiv.org/abs/1208.3922.

[35] W. Deng and W. Yin, "On the global and linear convergence of the generalized alternating direction method of multipliers," 2012. Rice University CAAM Technical Report TR12-14.

[36] V. Nedelcu, I. Necoara, and Q. T. Dinh, "Computational complexity of inexact gradient augmented Lagrangian methods: Application to constrained MPC," 2013. available at: arxiv.org/abs/1302.4355.

[37] D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation*. New Jersey: Prentice-Hall, Englewood Cliffs, 1989.

[38] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Cooperative convex optimization in networked systems: Augmented Lagrangian algorithms with directed gossip communication," *IEEE Transactions on Signal Processing*, vol. 59, pp. 3889–3902, August 2011.

[39] J. Mota, J. Xavier, P. Aguiar, and M. Pueschel, "Basis pursuit in sensor networks," in *ICASSP '11, IEEE International Conference on Acoustics, Speech, and Signal Processing*, (Prague, Czech Republic), pp. 2916–2919, May 2011.

[40] J. Mota, J. Xavier, P. Aguiar, and M. Pueschel, "Distributed basis pursuit," *IEEE Trans. Sig. Process.*, vol. 60, pp. 1942–1956, July 2012.

[41] H. Terelius, U. Topcu, and R. M. Murray, "Decentralized multi-agent optimization via dual decomposition," in *18th World Congress of the International Federation of Automatic Control (IFAC)*, (Milano, Italy), August 2011. identifier: 10.3182/20110828-6-IT-1002.01959.

[42] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSNs with noisy links – Part I: Distributed estimation of deterministic signals," *IEEE Trans. Sig. Process.*, vol. 56, pp. 350–364, Jan. 2009.

[43] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSNs with noisy links – Part I: Distributed estimation and smoothing of random signals," *IEEE Trans. Sig. Process.*, vol. 56, pp. 1650–1666, Jan. 2009.

[44] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *CDC 2012, IEEE International Conference on Decision and Control*, (Maui, Hawaii), pp. 5445–5450, Dec. 2012.

[45] E. Ghadimi, M. Johansson, and I. Shames, "Accelerated gradient methods for networked optimization,"

[46] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Weight optimization for consenus algorithms with correlated switching topology," *IEEE Transactions on Signal Processing*, vol. 58, pp. 3788–3801, July 2010.

[47] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Convergence rates of distributed Nesterov-like gradient methods on random networks," *submitted to IEEE Transactions on Signal Processing*, May 2013.

[48] D. Jakovetic, J. M. F. Moura, and J. Xavier, "Linear convergence rates of a class of distributed augmented Lagrangian algorithms," May 2013. to be submitted.

[49] D. Jakovetic, J. M. F. Moura, and J. Xavier, "Distributed Nesterov-like gradient algorithms," in *proc. CDC'12, 51st IEEE Conference on Decision and Control*, (Maui, Hawaii), pp. 5459–5464, Dec. 2012.

[50] D. Jakovetic, J. M. F. Moura, and J. Xavier, "Fast cooperative distributed learning," in *proc. Asilomar Conference on Signals, Systems, and Computers*, (Pacific Grove, CA), pp. 1513–1517, Nov. 2012.

[51] D. Bajovic, D. Jakovetic, J. M. F. Moura, J. Xavier, and B. Sinopoli, "Large deviations performance of consensus+innovations detection with non-Gaussian observations," *IEEE Transactions on Signal Processing*, vol. 60, pp. 5987–6002, November 2012.

[52] D. Bajovic, D. Jakovetic, J. Xavier, B. Sinopoli, and J. M. F. Moura, "Distributed detection via Gaussian running consensus: Large deviations asymptotic analysis," *IEEE Transactions on Signal Processing*, vol. 59, pp. 4381–4396, September 2011.

[53] D. Jakovetic, J. M. F. Moura, and J. Xavier, "Consensus+innovations detection: Phase transition under communication noise," in *proc. Allerton Conference on Communication, Control, and Computing*, (Monticello, IL), pp. 5987–6002, Oct. 2012.

[54] D. Bajovic, D. Jakovetic, J. M. F. Moura, J. Xavier, and B. Sinopoli, "Large deviations analysis of consensus+innovations detection in random networks," in *49th Allerton Conference of Communication, Control, and Computing*, (Monticello, IL), pp. 151–155, Oct. 2011.

[55] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Convergence rate analysis of distributed gradient methods for smooth optimization," in *proc. TELFOR 2012, 20th Telecommunications Forum*, (Belgrade, Serbia), pp. 867–870, Nov. 2012.

[56] D. Bajovic, D. Jakovetic, J. Xavier, B. Sinopoli, and J. M. F. Moura, "Distributed detection over time varying networks: Large deviations analysis," in *48th Allerton Conference of Communication, Control, and Computing*, (Monticello, Illinois), pp. 302–309, Oct. 2010.

[57] D. Bajovic, D. Jakovetic, J. Xavier, B. Sinopoli, and J. M. F. Moura, "Asymptotic performance of distributed detection over random networks," in *ICASSP '11, IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3008–3011, May 2011.

[58] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Consensus in correlated random topologies: Weights for finite time horizon," in *proc. ICASSP 2010, IEEE International Conference on Acoustics, Speech and Signal Processing*, (Dallas, TX), pp. 2974–2977, March 2010.

[59] M. Schmidt, N. L. Roux, and F. Bach, "Convergence rates of inexact proximal-gradient methods for convex optimization," in *Neural Information Processing systems (NIPS)*, (Granada, Spain), December 2011.

[60] O. Devolder, F. Glineur, and Y. Nesterov, "First-order methods of smooth convex optimization with inexact oracle," *submitted to Mathematical Programming*, 2011. available at: http://www.optimization-online.org/DB_FILE/2010/12/2865.pdf.

[61] M. H. DeGroot, "Reaching a consensus," *Journal of the American Statistical Association*, vol. 69, pp. 118–121, March 1974. Theory and Methods Section.

[62] P. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *Journal of Machine Learning Research*, vol. 11, pp. 1663–1707, May 2010.

[63] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[64] D. Blatt and A. O. Hero, "Energy based sensor network source localization via projection onto convex sets (POCS)," *IEEE Trans. on Signal Processing*, vol. 54, no. 9, pp. 3614–3619, 2006.

[65] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *IPSN '05, Information Processing in Sensor Networks*, (Los Angeles, California), pp. 63–70, 2005.

[66] D. Blatt, A. Hero, and H. Gauchman, "A convergent incremental gradient method with a constant step size," *Siam J. Optim.*, vol. 18, no. 1, pp. 29–51, 2009.

[67] S. Richter, "Computational complexity certi

cation of gradient methods for real-time model predictive control," 2012. PhD Thesis, ETH Zurich.

[68] B. T. Polyak, *Introduction to Optimization*. Optimization Software, 1987.

[69] Y. Nesterov, "On an approach to the construction of optimal methods of minimization of smooth convex functions," *Ekonomika i Mateaticheskie Metody*, vol. 24, pp. 509–517, 1988.

[70] Y. Nesterov, "Smooth minimization of nonsmooth functions," *Mathematical programming*, vol. 103, no. 1, pp. 127–152, 2005.

[71] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imaging Sciences*, vol. 2, no. 2, pp. 183–202, 2009.

[72] P. Tseng, "On accelerated proximal gradient methods for convex-concave optimization," *submitted to SIAM Journal on Optimization*, 2008.

[73] L. Vandenberghe, "Optimization methods for large-scale systems," 2010. Lecture notes, available at: http://www.ee.ucla.edu/ vandenbe/ ee236c.html.

[74] G. Shi and K. H. Johansson, "Finite-time and asymptotic convergence of distributed averaging and maximizing algorithms," 2012. available at: http://arxiv.org/pdf/1205.1733.pdf.

[75] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," in *36th Annual ACM Symposium on Theory of Computing*, (Chicago, IL), pp. 561–568, August 2004.

[76] M. Zargham, A. Ribeiro, and A. Jadbabaie, "A distributed line search for network optimization," in *American Control Conference*, (San Francisco, CA), pp. 472–477, June 2012.

[77] A. Tahbaz-Salehi and A. Jadbabaie, "Gradient methods for minimizing composite objective function," 2007. Technical Report 76, Center for Operations Research and Econometrics (CORE), Catholic University of Louvain (UCL).

[78] 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, Michael Jordan, Editor in Chief*, vol. 3, no. 1, pp. 1–122, 2011.

[79] A. Dimakis, S. Kar, J. M. F. Moura, M. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1847–1864, 2010.

[80] A. Tahbaz-Salehi and A. Jadbabaie, "On consensus in random networks," in *44th Annual Allerton Conference on Communication, Control, and Computing*, (Monticello, Illinois, USA), pp. 1315–1321, September 2006.

[81] A. Nedic, "Optimization I." lecture notes, August 2008.

[82] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, 2004.

[83] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *in Proc. 1st Int. ACM Conf. Embedded Networked Sensor Systems*, (Los Angeles, CA), pp. 1–13, Nov. 2003.

[84] A. Cerpa, J. Wong, L. Kuang, M. Potkonjak, and D. Estrin, "Understanding packet delivery performance in dense wireless sensor networks," in *in Proc. 4th Int. Symp. Information Processing in Sensor Networks (IPSN)*, (Los Angeles, CA), pp. 81–88, Apr. 2005.

[85] P. Denantes, F. Benezit, P. Thiran, and M. Vetterli, "Which distributed averaging algorithm should i choose for my sensor network," in *in Proc. 7th IEEE Conf. Computer Communications (INFOCOM)*, (Phoenix, Arizona), pp. 986–994, March 2008.

[86] W. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, Apr. 1970.

[87] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing markov chain on a graph," *SIAM Review*, vol. 46, pp. 667–689, Dec. 2004.

[88] A. Tahbaz-Salehi and A. Jadbabaie, "Consensus over ergodic stationary graph processes," *IEEE Trans. Autom. Control*, vol. 55, pp. 225–230, Jan. 2010.

[89] A. F. Karr, *Probability Theory*. ser. Springer Texts in Statistics, New York: Springer-Verlag, 1993.

[90] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, UK: Cambridge Univ. Press, 1990.

[91] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Control Lett.*, vol. 53, pp. 65–78, Sep. 2004.

[92] J.-B. H. Urruty and C. Lemarechal, *Convex Analysis and Minimization Algorithms I: Fundamentals*. Springer Verlag, 1996.

[93] B. Quaqish, "A family of multivariate binary distributions for simulating correlated binary variables with specified marginal means and correlations," *Biometrika*, vol. 90, pp. 455–463, June 2003.

[94] S. D. Oman and D. M. Zucker, "Modelling and generating correlated binary variables," *Biometrika*, vol. 88, pp. 287–290, March 2001.

[95] R. T. Rockafellar, "The multiplier method of Hestenes and Powell applied to convex programming," *Journal on Optimization Theory and Applications*, vol. 12, no. 6, pp. 133–145, 1973.

[96] D. Bertsekas, "Multiplier methods: A survey," *Automatica*, vol. 12, pp. 133–145, 1976.

[97] B. He and X. Yuan, "On the acceleration of augmented Lagrangian method for linearly constrained optimization," 2010. available at: http://www.optimization-online.org/DB_HTML/2010/10/2760.html.

[98] B. Johansson, M. Rabi, and M. Johansson, "A randomized incremental subgradient method for distributed optimization in networked systems," *SIAM Journal on Optimization*, vol. 20, pp. 1157–1170, August 2009.

[99] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[100] X. Hong, C. Wang, H. Chen, and Y. Zhang, "Secondary spectrum access networks," *IEEE Veh. Technol. Mag.*, vol. 4, no. 2, pp. 36–43, 2009.

[101] B. Gharesifard and J. Cortes, "Distributed continuous-time convex optimization on weighted-balanced digraphs," 2012. available at: arxiv.org/abs/1204.0304.

[102] J. Wang and N. Elia, "Control approach to distributed optimization," in *48th Allerton Conference on Communication, Control, and Computing*, (Monticello, IL), pp. 557–561, Oct. 2010.

[103] J. Wang and N. Elia, "A control perspective to centralized and distributed convex optimization," in *50th CDC Conference on Decision and Control*, (Orlando, Florida), pp. 3800–3805, Dec. 2011.

[104] A. Nedic and A. Ozdaglar, "Subgradient methods for saddle point problems," *Journal of Optimization Theory and Applications*, vol. 142, no. 1, pp. 205–208, 2009.

[105] T. Erseghe, D. Zennaro, E. Dall'Anese, and L. Vangelista, "Fast consensus by the alternating direction multipliers method," *IEEE Trans. Sig. Process.*, vol. 59, pp. 5523–5537, Nov. 2011.

[106] F. Chung, *Spectral graph theory*. American Mathematical Soc., 1997.

[107] J. E. Marsden and A. J. Tromba, *Vector Calculus*. Freeman and Company, New York, 1996.